Introduction to Convolutional Neural Network

Prof. Chia-Yu Lin

Yuan Ze University

2021 Spring

Thanks to the slides from Prof. Hung-Yi Lee in NTU.

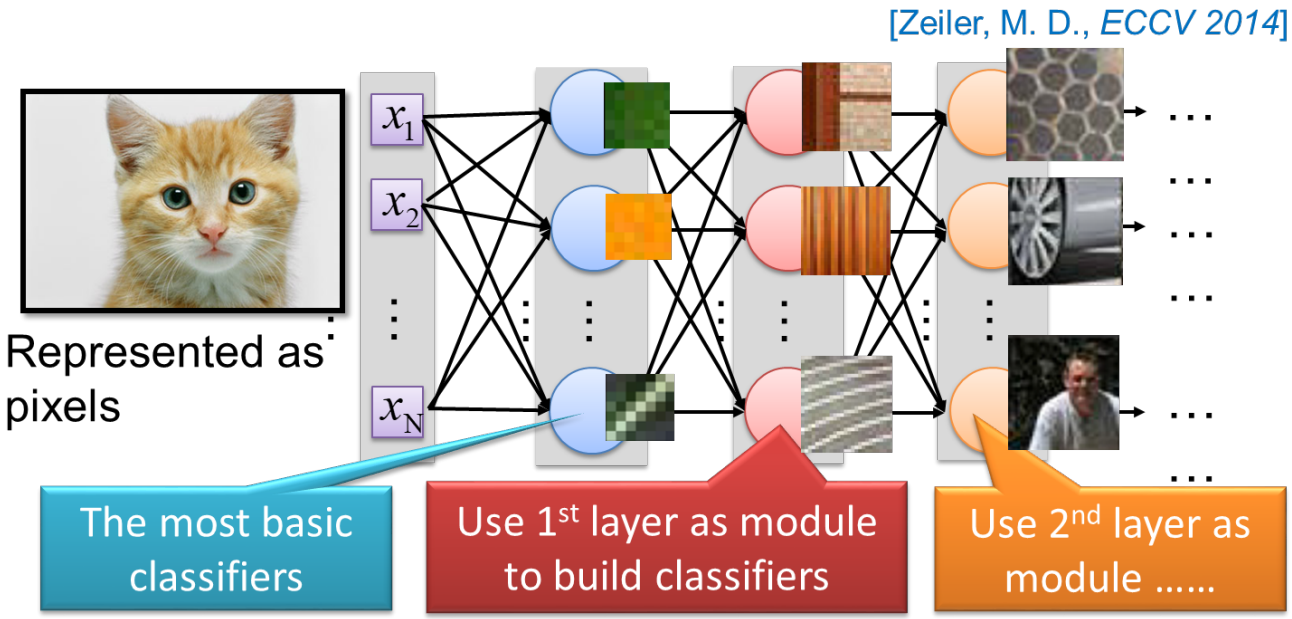
Outline

- Industry 4.0 problem
- Why CNN for image?
- Pixels in image
- CNN model
  - Convolution layer
  - Max Pooling layer
  - Flatten
- Famous CNN model
- What does CNN learn?
- CNN Application
- CNN for Industry 4.0
- Summary

Why CNN for image?

Feed Image into DNN

- Put whole image into DNN is inefficient.
- Can the network be simplified by considering the properties of images?



History of CNN

- CNN是1960年代，由美國兩位生物學家Hubel、Wiesel在研究貓腦皮層時，發現局部敏感和方向選擇的神經元，具有其獨特的網絡結構可以有效地降低反饋神經網絡的複雜性。
- 1980年 Kunihiko Fukushima 提出了CNN的前身——neocognitron。
- 20世紀 90 年代，LeCun et al. [3] 等人發表論文，確立了CNN的現代結構，後來又對其進行完善。
  - 他們設計了一種多層的人工神經網路，取名叫做LeNet-5，可以對手寫數字做分類。和其他神經網路一樣，LeNet-5 也能使用 backpropagation 演算法訓練。

https://codertw.com/%E7%A8%8B%E5%BC%8F%E8%AA%9E%E8%A8%80/500308/

Why CNN for Image

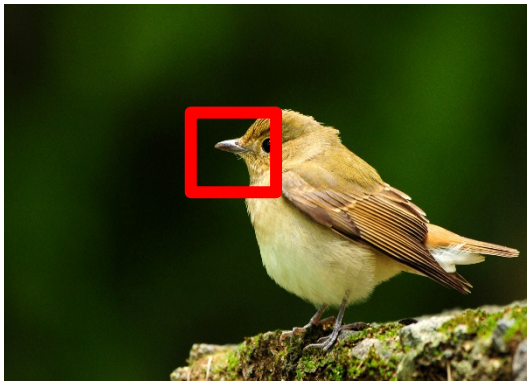
- Some patterns are much smaller than the whole image

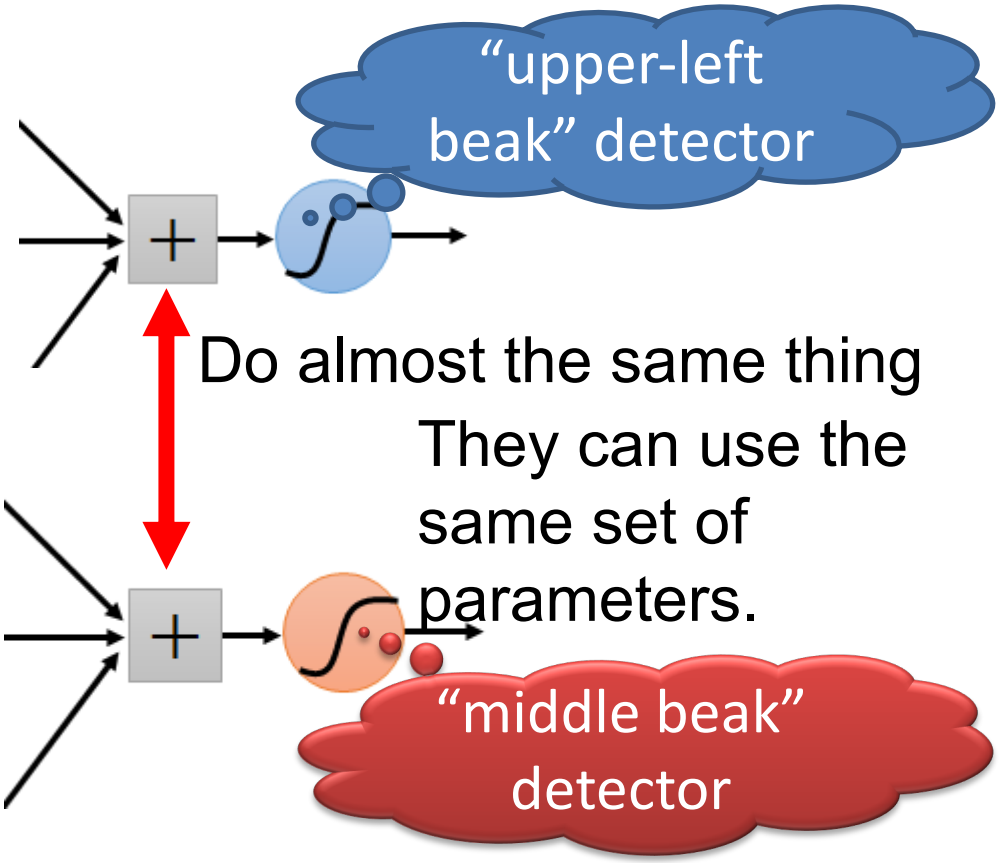
A neuron does not have to see the whole image to discover the pattern.

Connecting to small region with less parameters

Why CNN for Image

- The same patterns appear in different regions.





Why CNN for Image

- Subsampling the pixels will not change the object.

bird

bird

subsampling

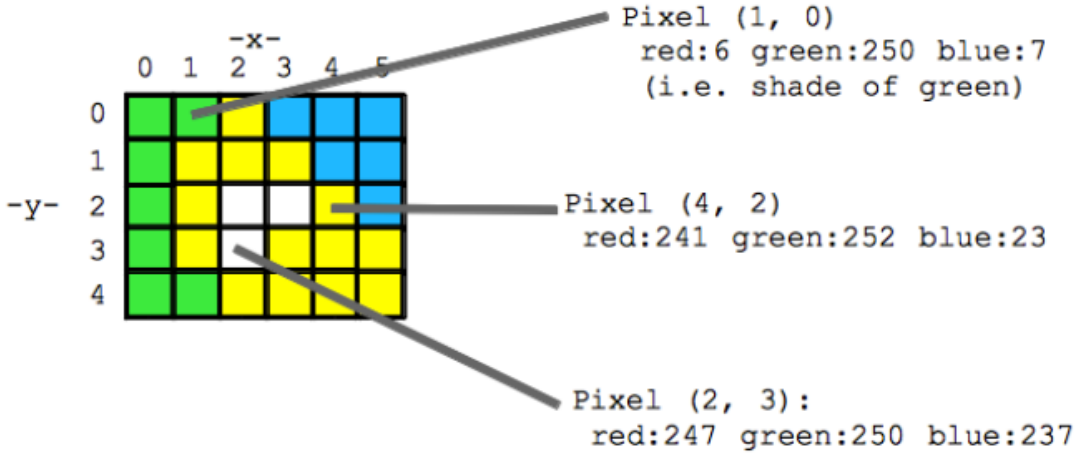
We can subsample the pixels to make image smaller

Less parameters for the network to process the image

Pixels in Image

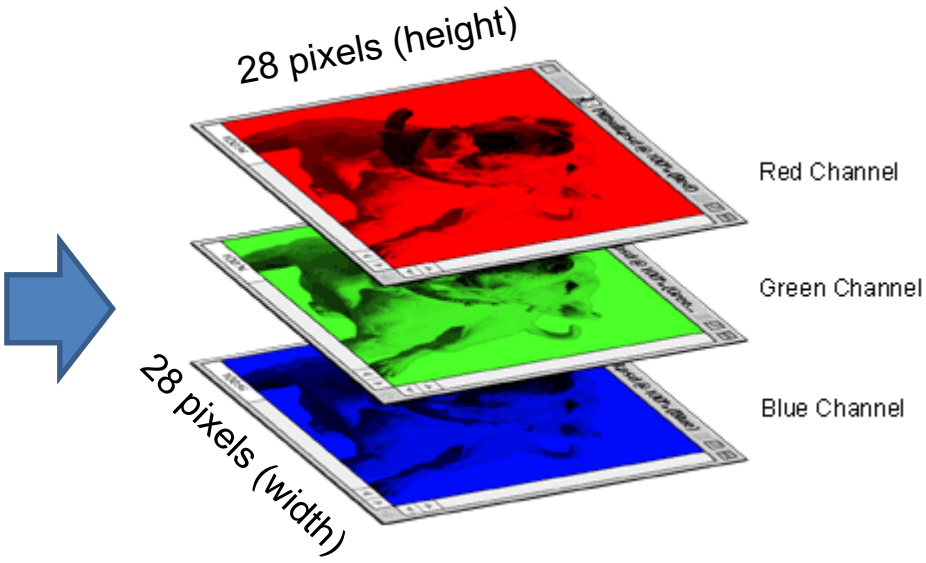
Pixels in Image (1/4)

- Each image contain many pixels.
- Each pixels compose 3 channels
  - red, green, blue (RGB).
- Each channel have brightness levels between 0~255.



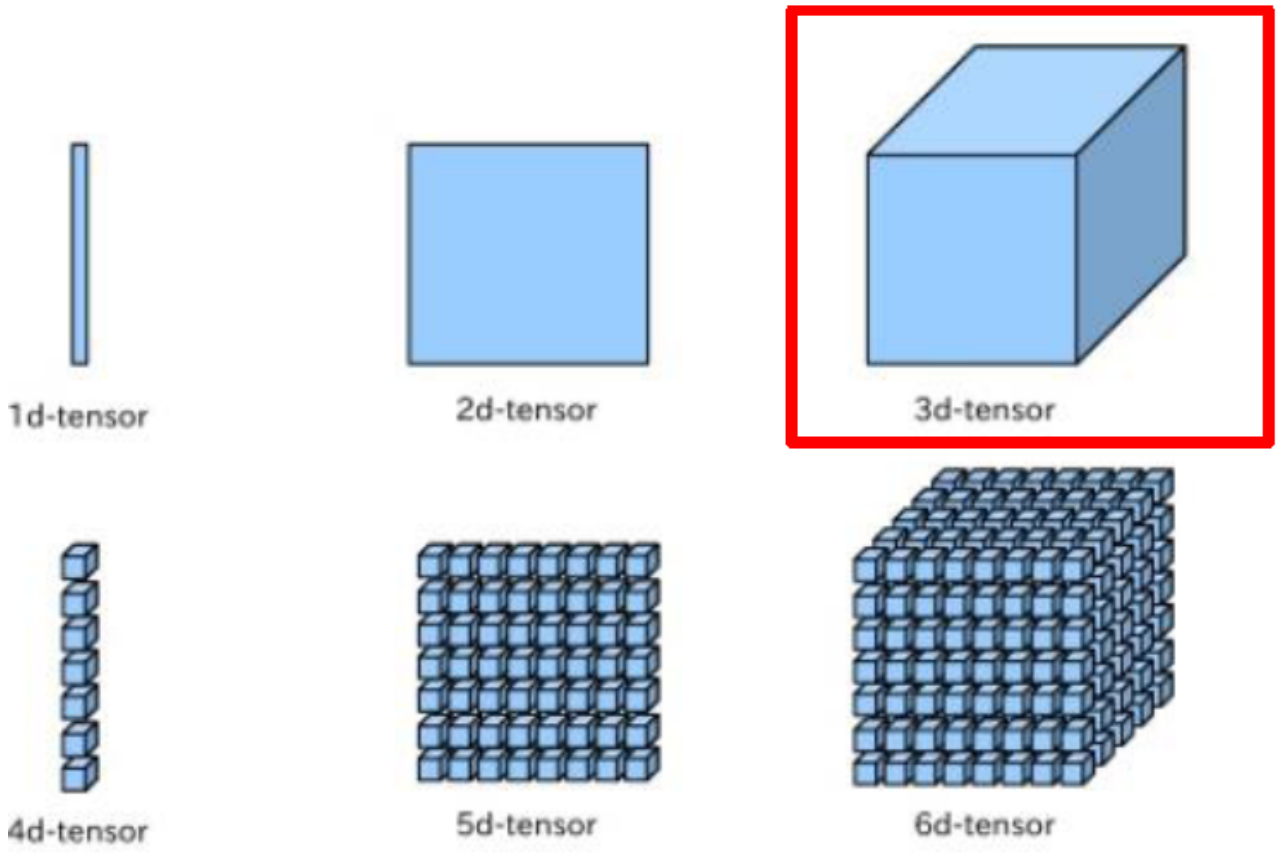
Pixels in Image(2/4)

Color Image
(RGB)

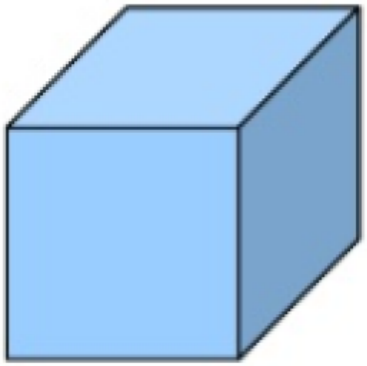


3 Channels
(RGB)

Pixels in Image(3/4)



Pixels in Image(4/4)



= [image width, image height, image channel(feature map)]

(R,G,B)

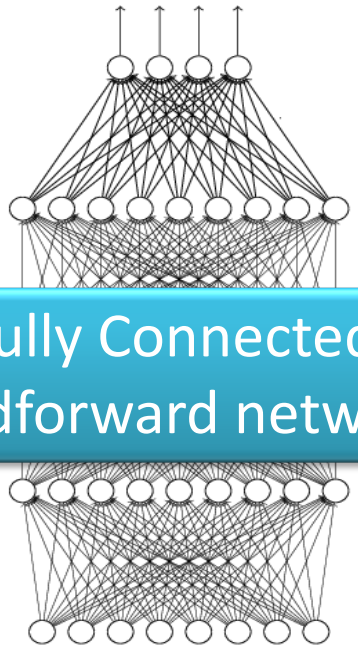
A image is a 3D-tensor

Input many images into a batch => Becomes a 4D-tensor

CNN Model

The whole

cat dog ……



Fully Connected Feedforward network

Convolution

Max Pooling

Convolution

Max Pooling

Can repeat many times

Flatten

The whole

Property 1

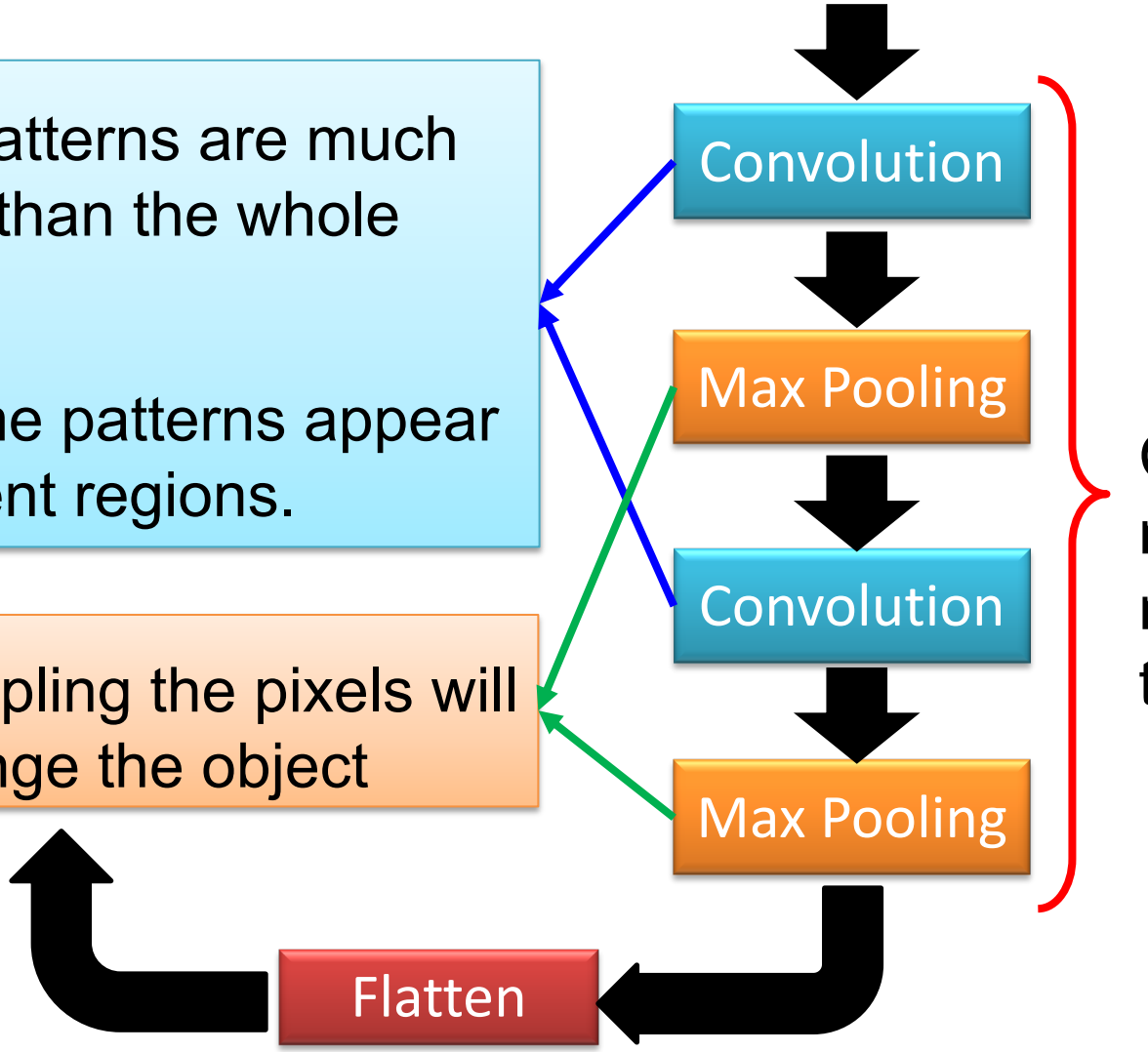
- Some patterns are much smaller than the whole

Property 2

- The same patterns appear in different regions.

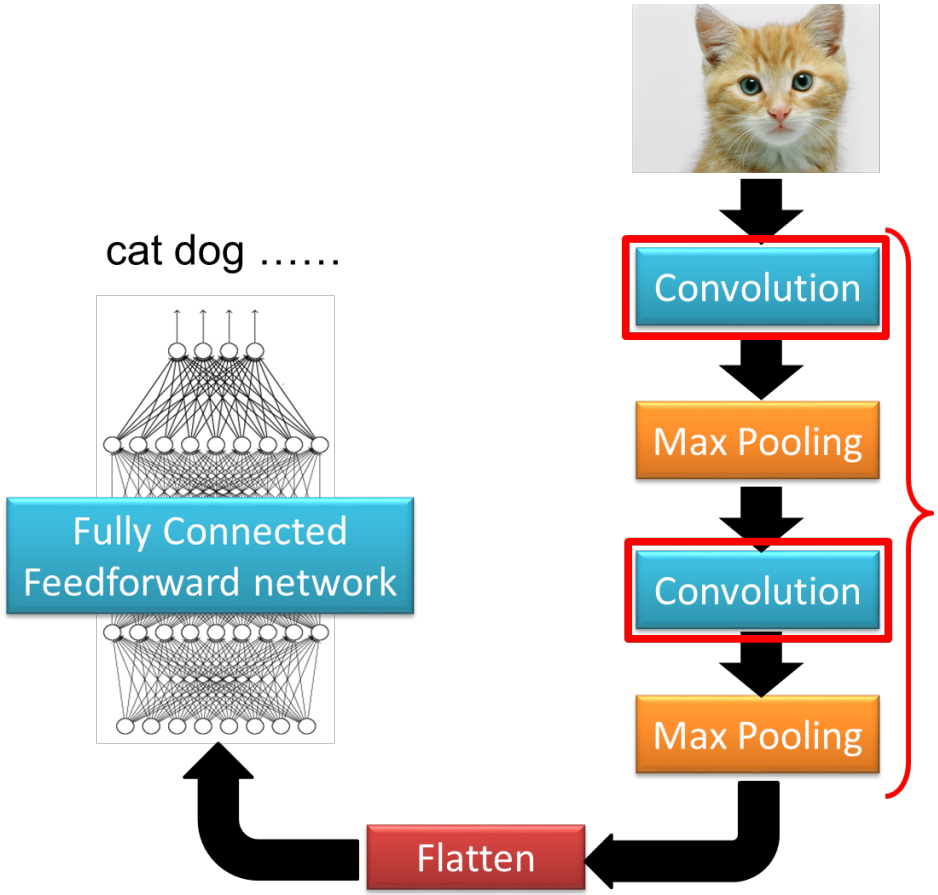
Property 3

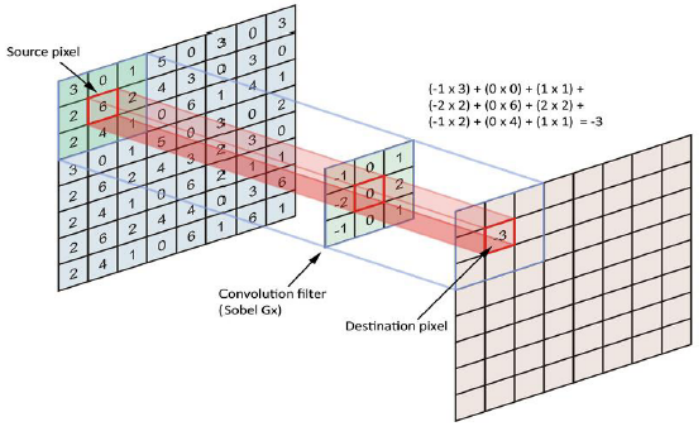
- Subsampling the pixels will not change the object



Convolution

- Extract some features on specific local area.





CNN – Convolution

Those are the network parameters to be learned.

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

Matrix

-1	1	-1
-1	1	-1
-1	1	-1

Filter 2

Matrix

⋮ ⋮

Property 1

Each filter detects a small pattern (3 x 3).

CNN – Convo

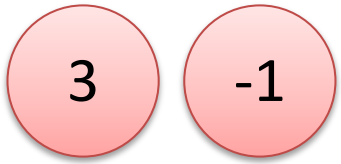
1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

stride=1

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image



CNN – Convo

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

If stride=2

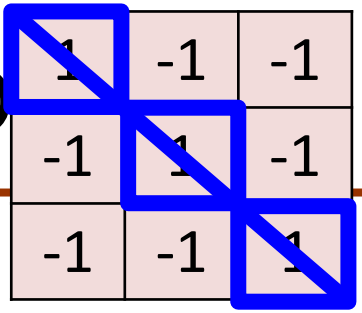
1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

3 -3

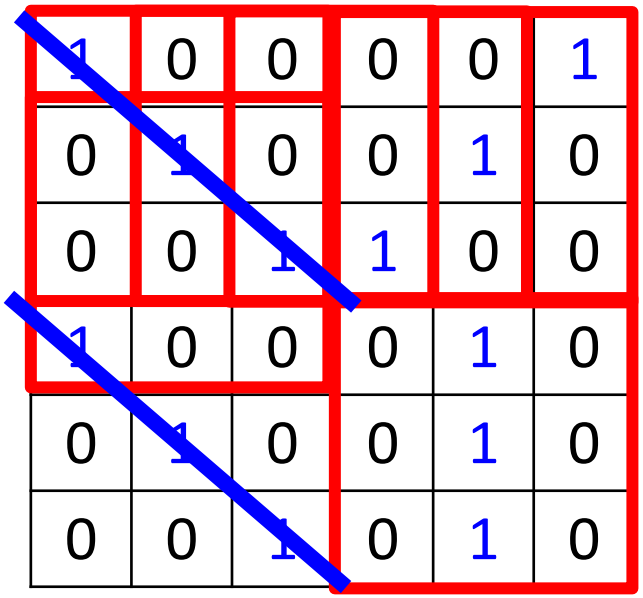
We set stride=1 below

CNN – Convo



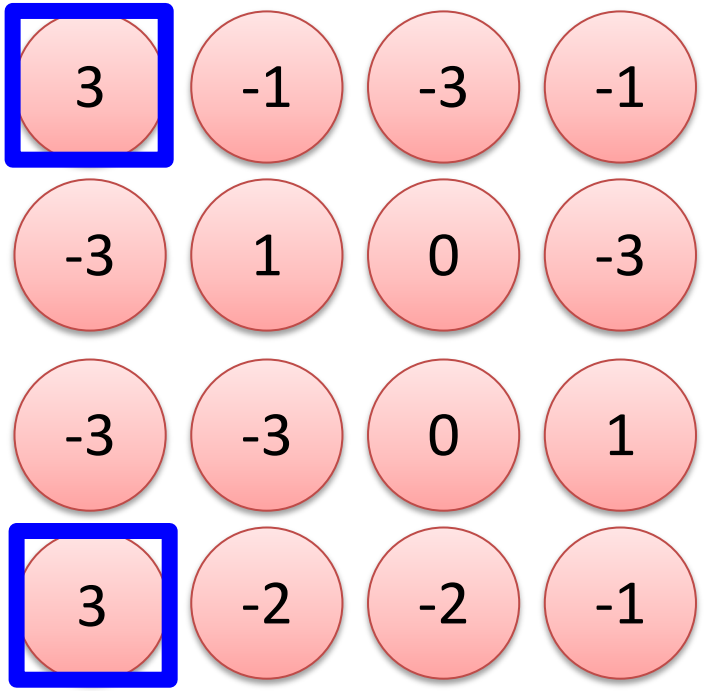
Filter 1

stride=1



6 x 6 image

Detect the pattern from west-north to east-south



The pattern fits filter.

Property 2

CNN – Convol

-1	1	-1
-1	1	-1
-1	1	-1

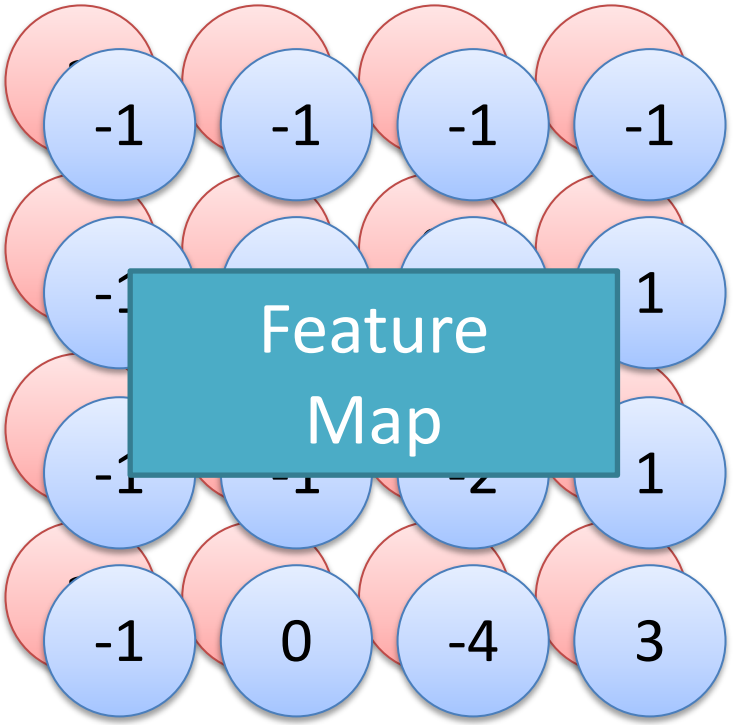
Filter 2

stride=1

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

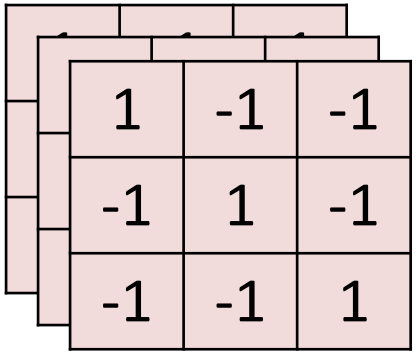
6 x 6 image

Do the same process for every filter

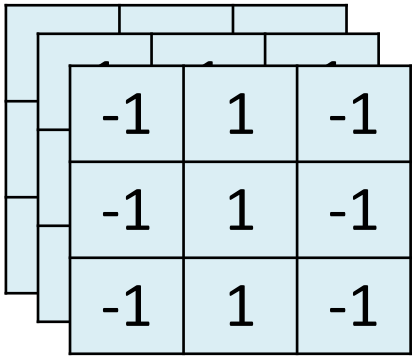


4 x 4 image

CNN – Colorful image (1/2)

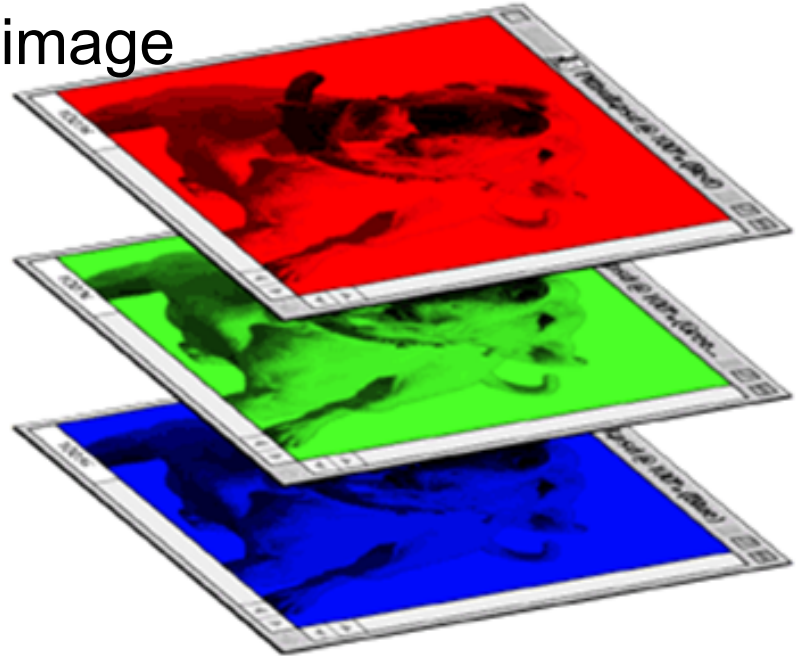


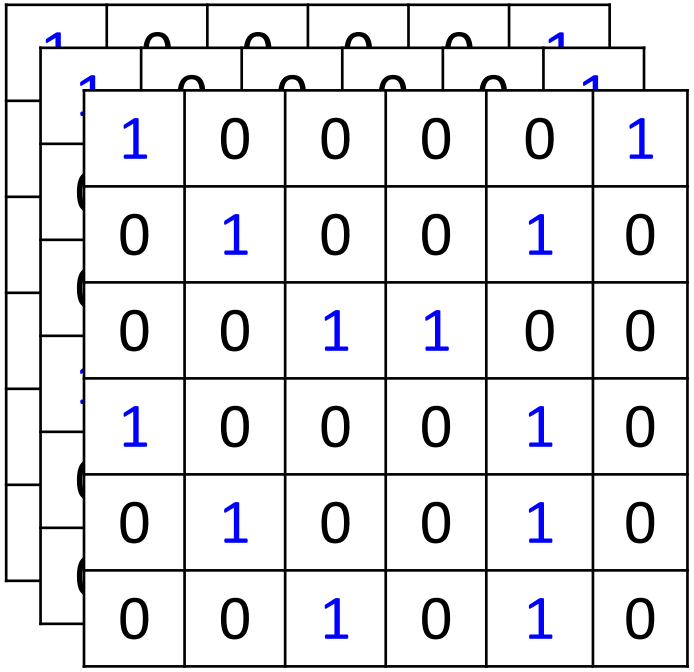
Filter 1



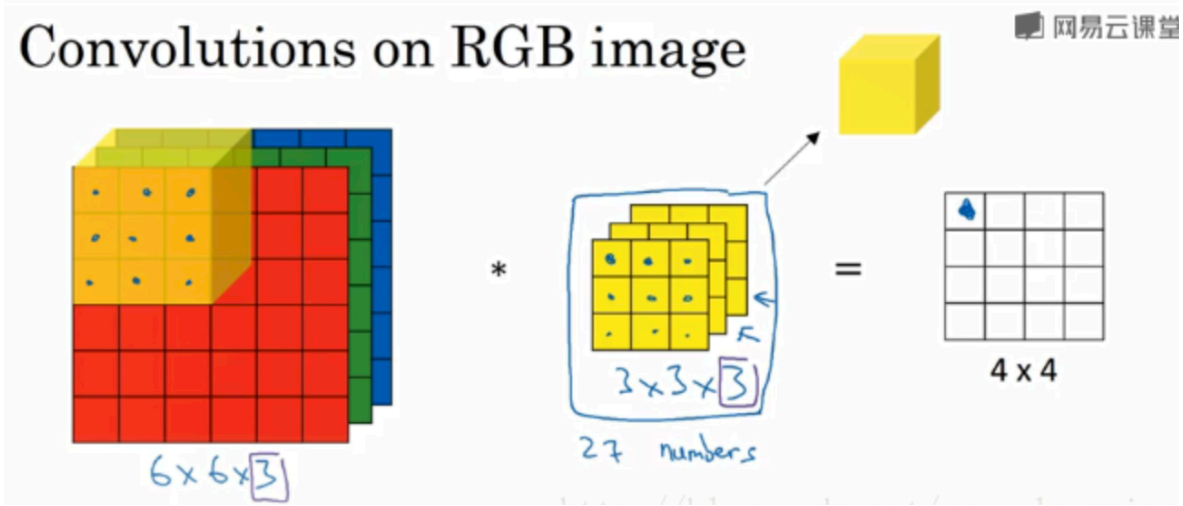
Filter 2

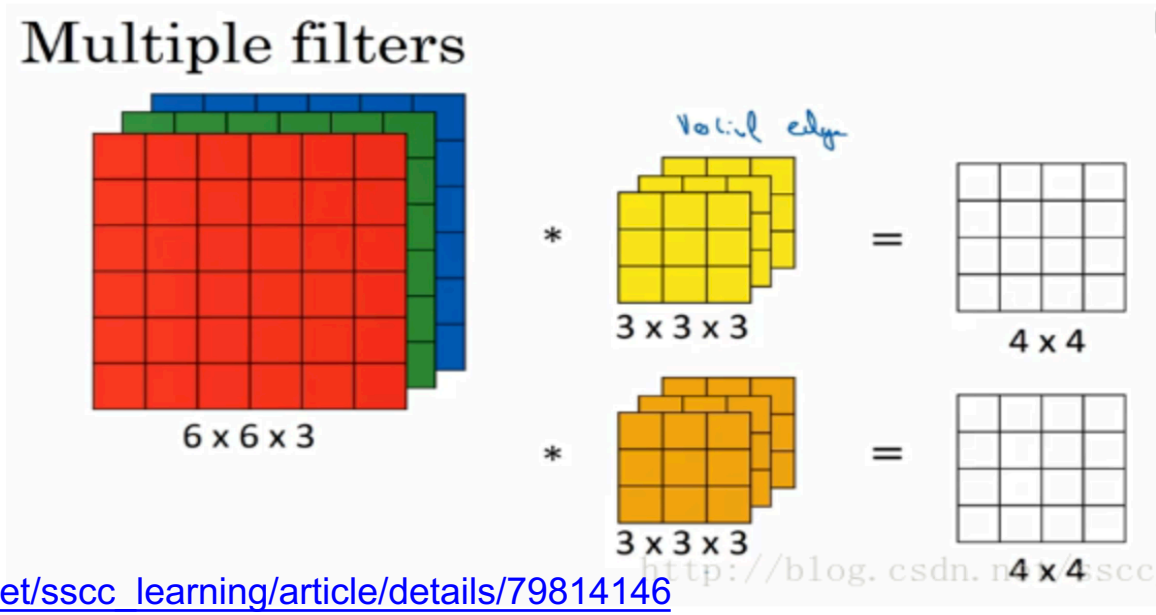
Colorful image





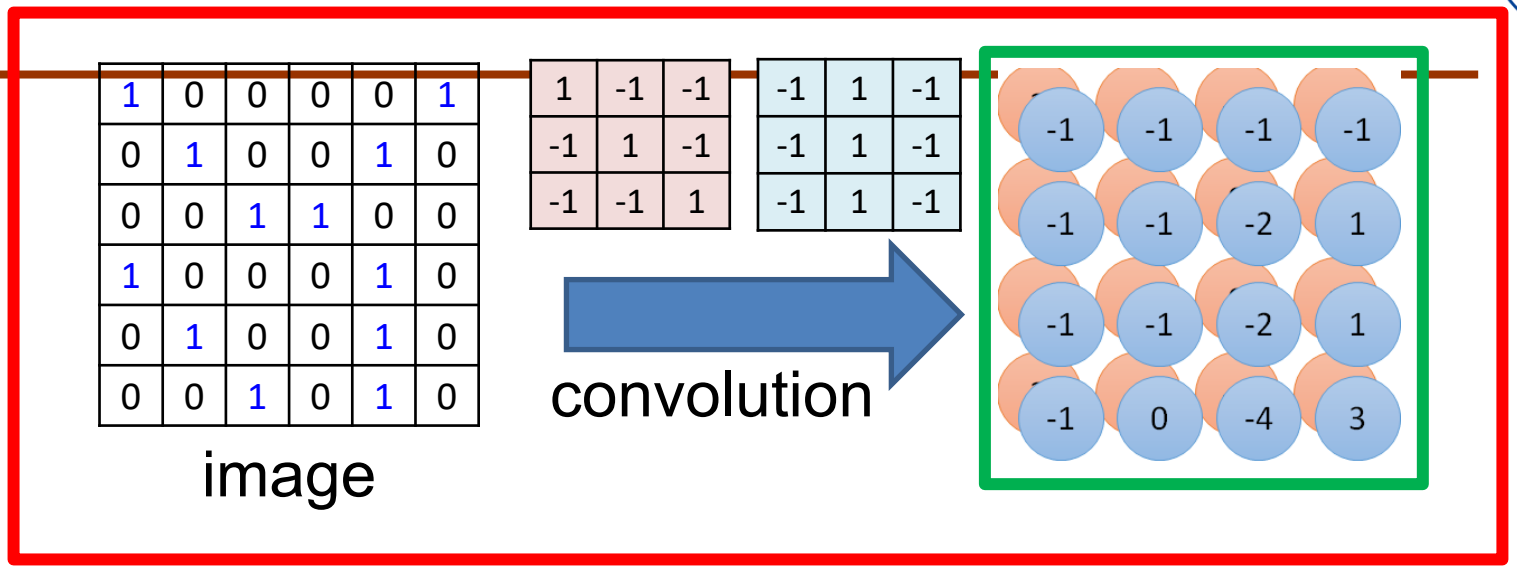
CNN – Colorful image (2/2)



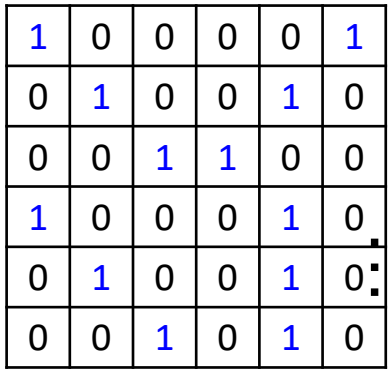


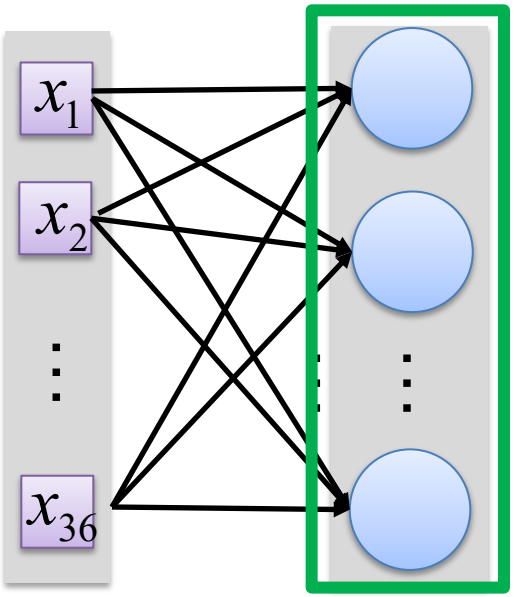
https://blog.csdn.net/sscc_learning/article/details/79814146

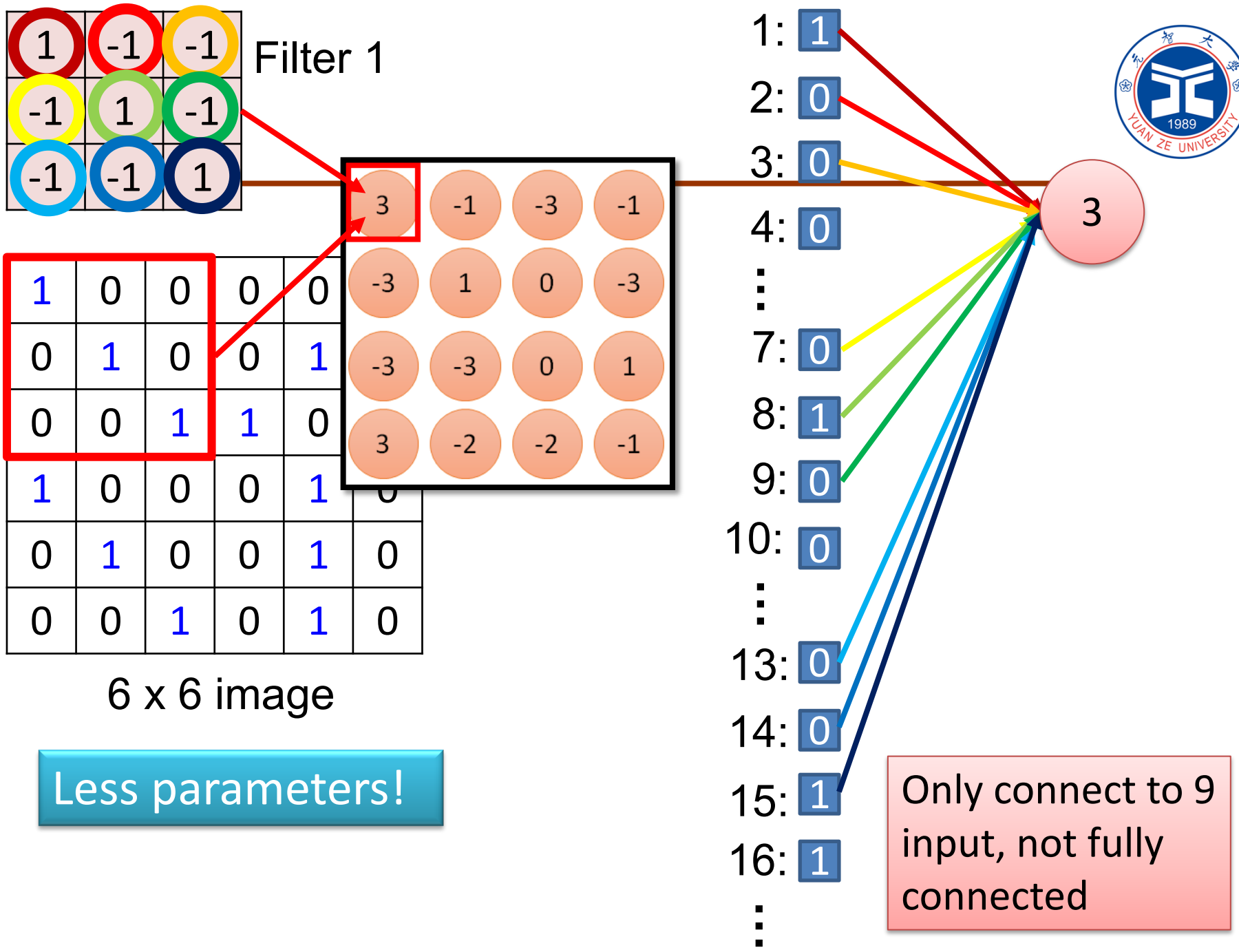
Convolution v.s. Fully Connected

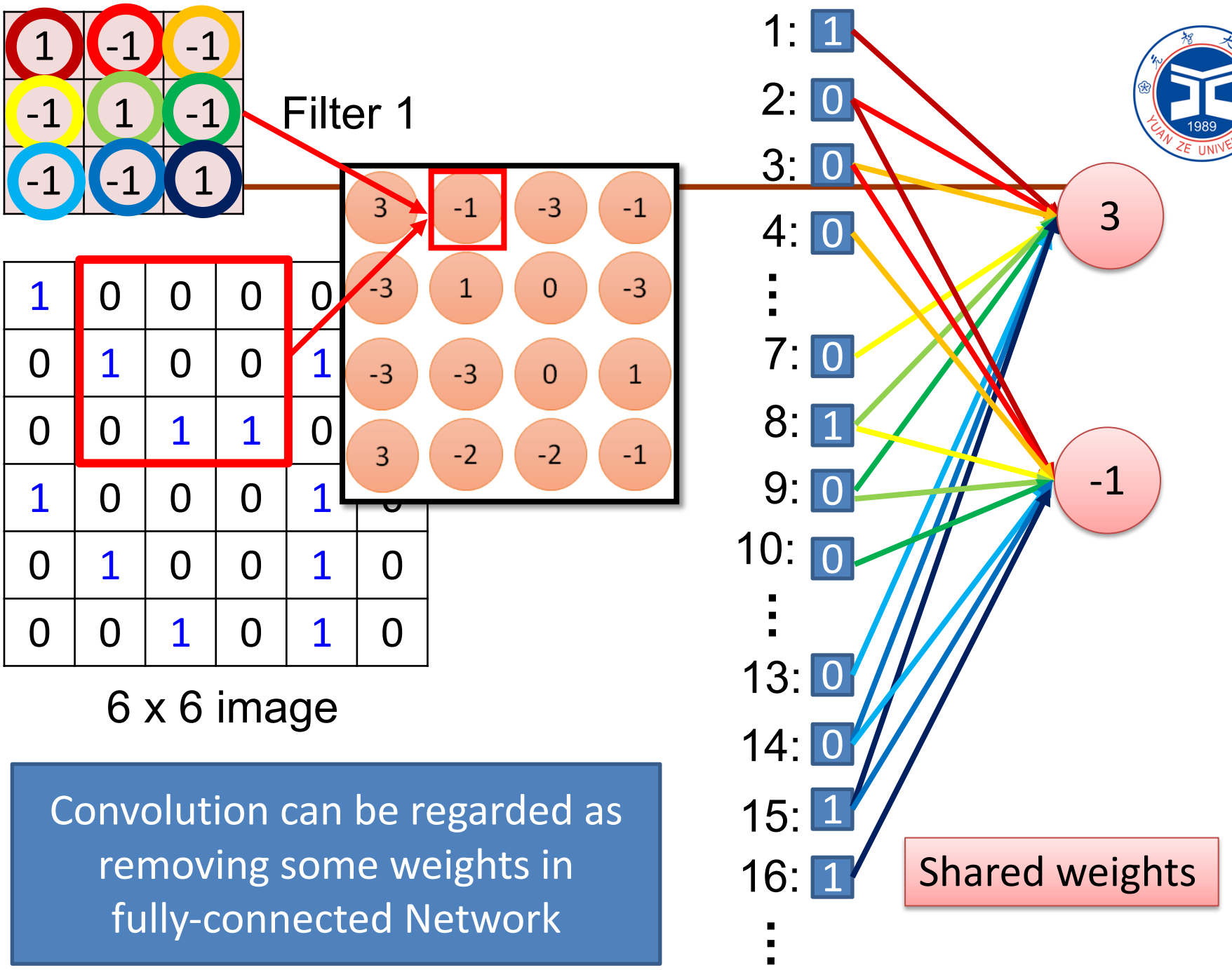


Fully-connected



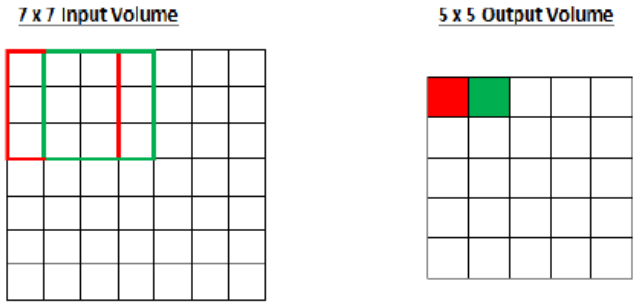



1 -1 -1
-1 1 -1
-1 -1 1
Filter 1
1 0 0 0 0
0 1 0 0 1
0 0 1 1 0
1 0 0 0 1
0 1 0 0 1 0
0 0 1 0 1 0
6 x 6 image
3 -1 -3 -1
-3 1 0 -3
-3 -3 0 1
3 -2 -2 -1
1: 1
2: 0
3: 0
4: 0
7: 0
8: 1
9: 0
10: 0
13: 0
14: 0
15: 1
16: 1
3
Less parameters!
Only connect to 9 input, not fully connected

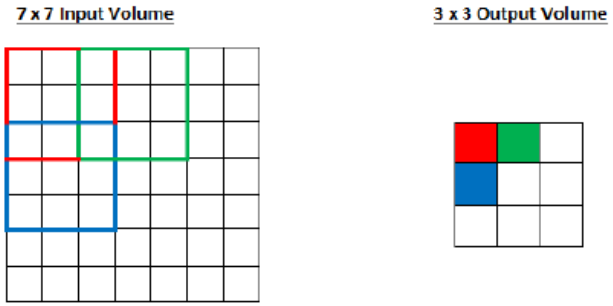

1 -1 -1
-1 1 -1
-1 -1 1
Filter 1
3 -1 -3 -1
-3 1 0 -3
-3 -3 0 1
3 -2 -2 -1
1 0 0 0 0
0 1 0 0 1
0 0 1 1 0
1 0 0 0 1
0 1 0 0 1 0
0 0 1 0 1 0
6 x 6 image
1: 1
2: 0
3: 0
4: 0
7: 0
8: 1
9: 0
10: 0
13: 0
14: 0
15: 1
16: 1
3
-1
Shared weights
Convolution can be regarded as removing some weights in fully-connected Network

Convolution Parameters (1/2)

- Filter size
- Stride
  - Amount of filter shift
- Padding
  - Padding zero on image boundary
  - Remain the same size after convolution



Size = 3*3 stride =1

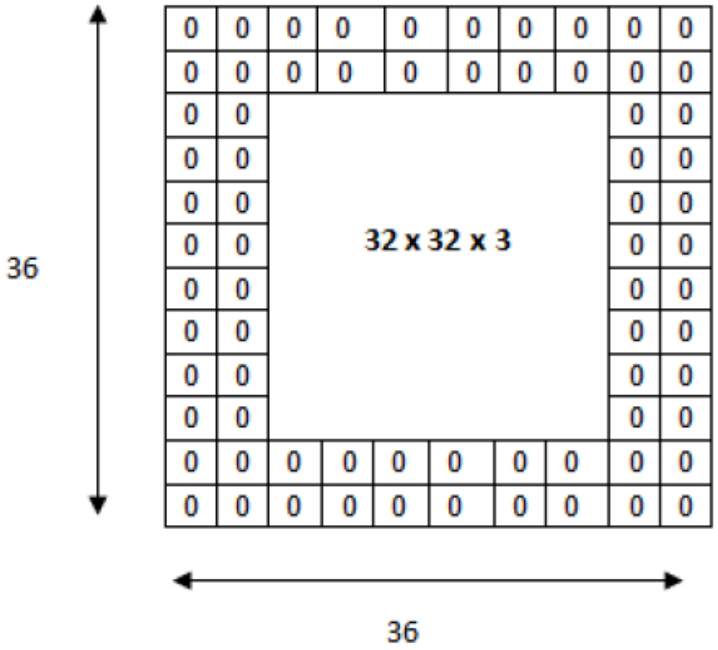


Size = 3*3 stride =2

Convolution Parameters (2/2)

- Filter size
- Stride
  - Amount of filter shift
- Padding
  - Padding zero on image boundary
  - Remain the same size after convolution

$$O = \frac{(W - K + 2P)}{S} + 1$$

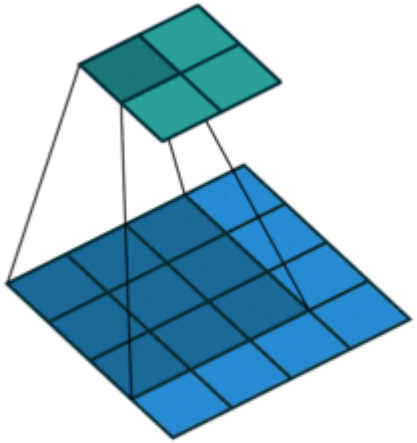


O: output height/length
W: input height/length
K: filter size
P : padding
S: stride size

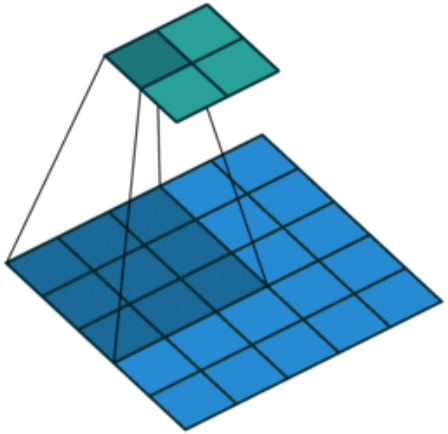
After 5*5 conv with stride 1 remain same size (padding=2)

Animation of Convolution

- https://github.com/vdumoulin/conv_arithmetic



No padding, Stride=1



No padding, Stride=2

Example-Convolution

0	0	0	0	0
0	0	-1	0	0
0	-1	5	-1	0
0	0	-1	0	0
0	0	0	0	0

Sharpen

Example-Convolution

0	0	0	0	0
0	1	1	1	0
0	1	1	1	0
0	1	1	1	0
0	0	0	0	0

Blur

Example-Convolution

	0	0	0	
	-1	1	0	
	0	0	0	

Edge enhance

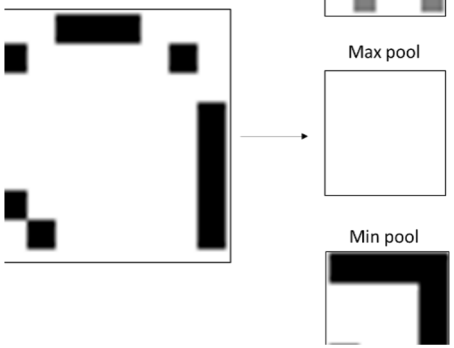
Example-Convolution

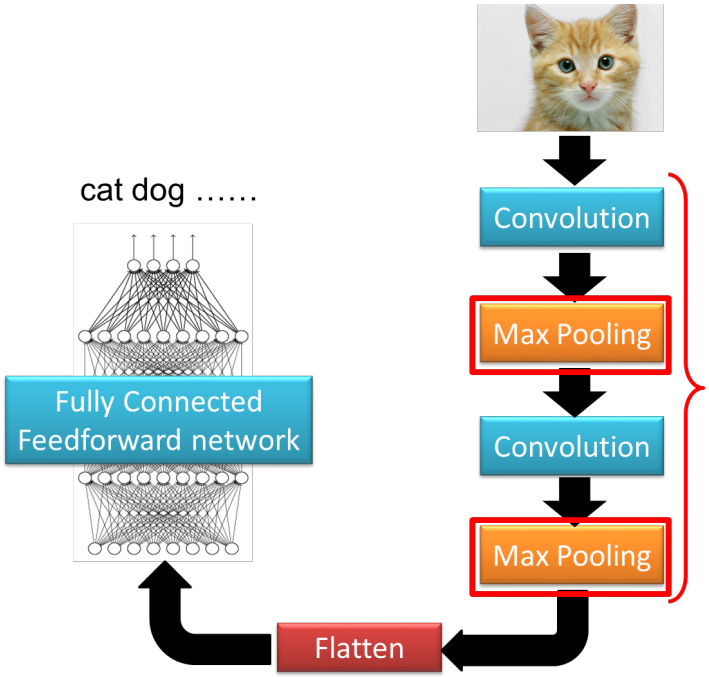
	0	1	0	
	1	-4	1	
	0	1	0	

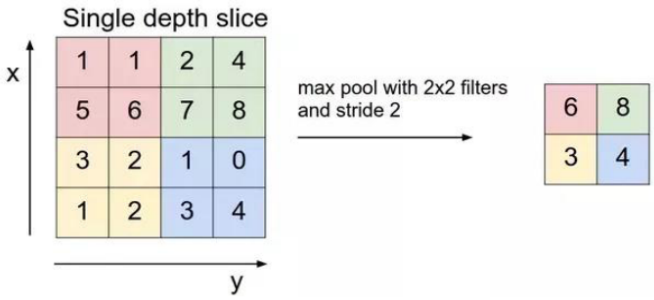
Edge detect

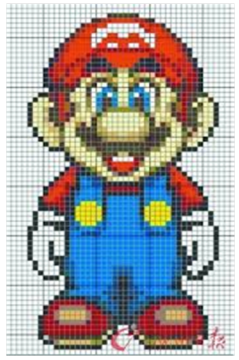
Max Pooling

- Subsampling the image
  - Make image smaller
  - Max pooling
  - Mean pooling
  - Min pooling









http://tech.sina.com.cn/it/2015-07-04/doc-ifxesftm9556659.shtml

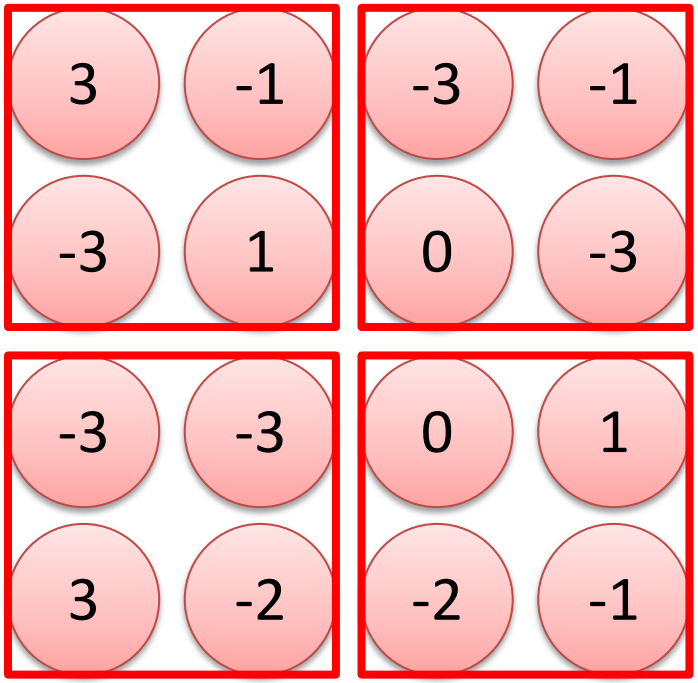
CNN – Max Pooling

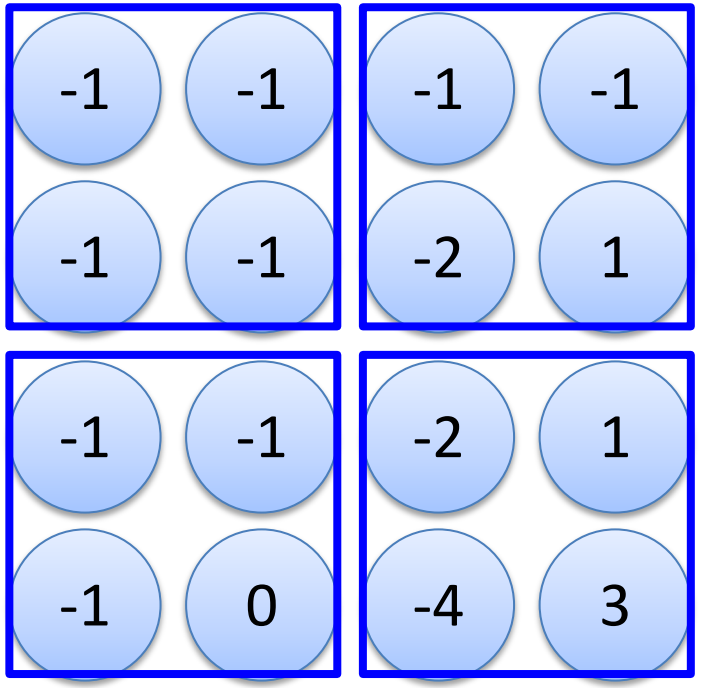
1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

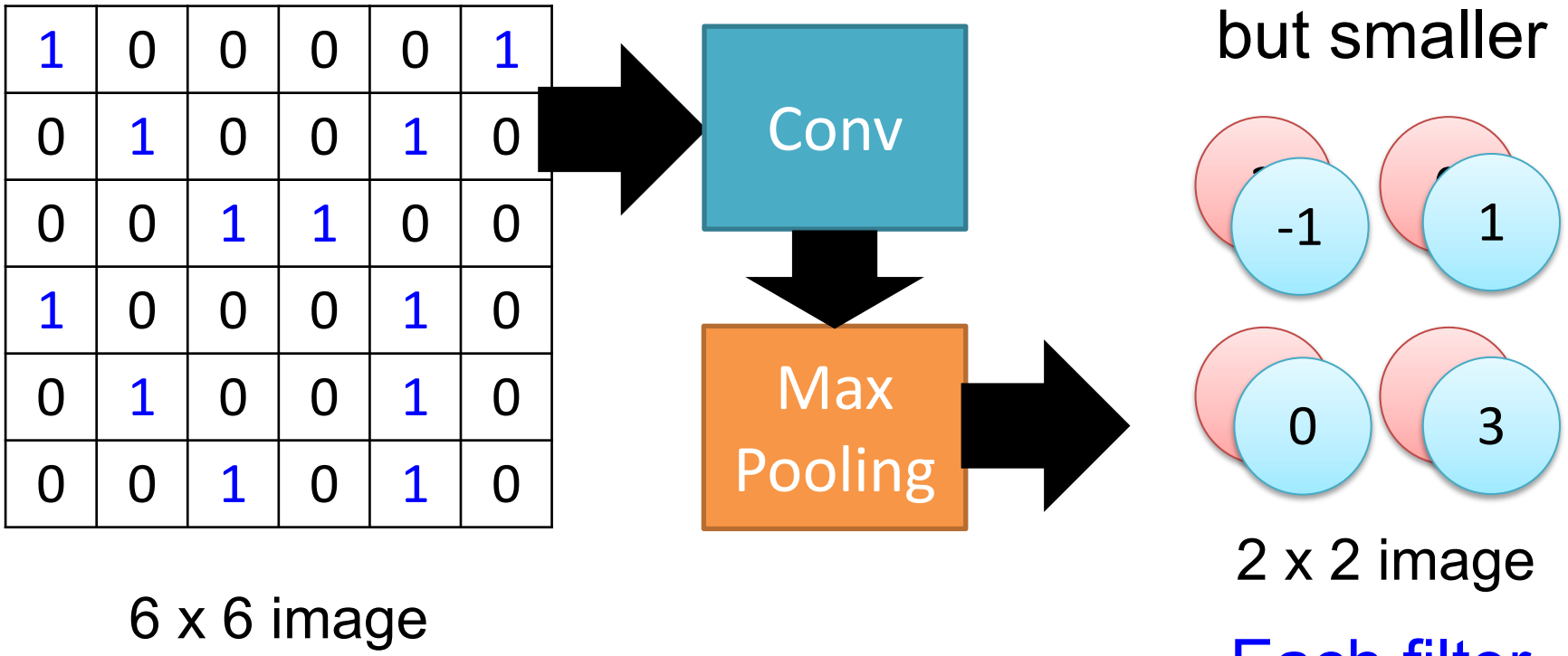
-1	1	-1
-1	1	-1
-1	1	-1

Filter 2

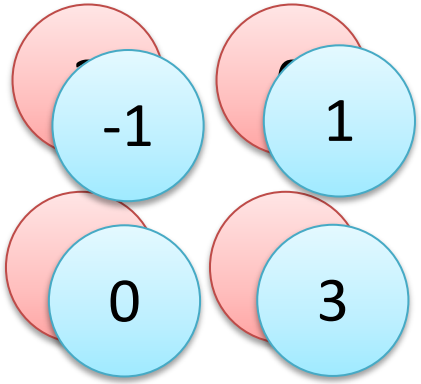




CNN – Max Pooling



The whole



A new image

Smaller than the original image

Convolution

Max Pooling

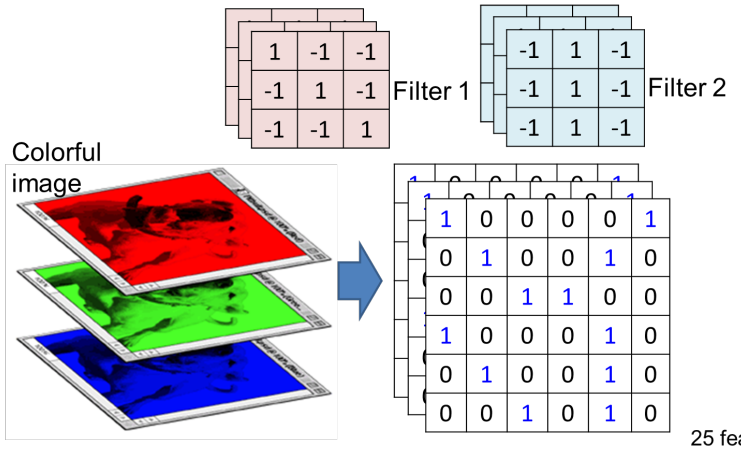
Convolution

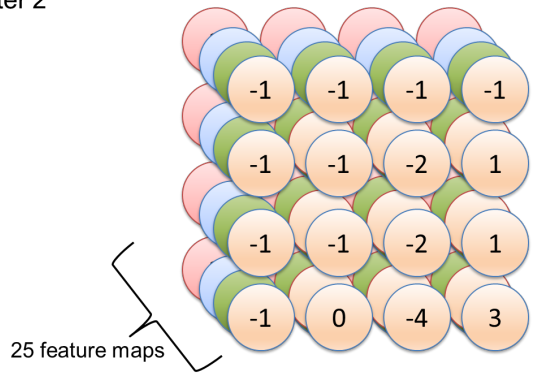
Max Pooling

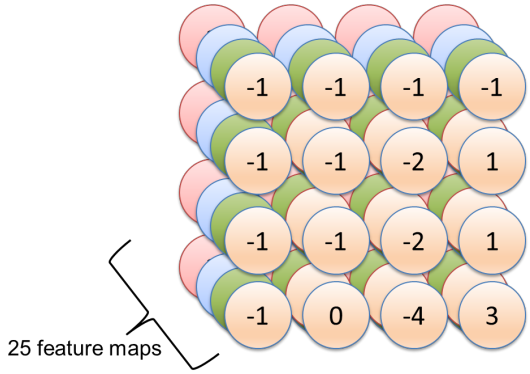
Can repeat many times

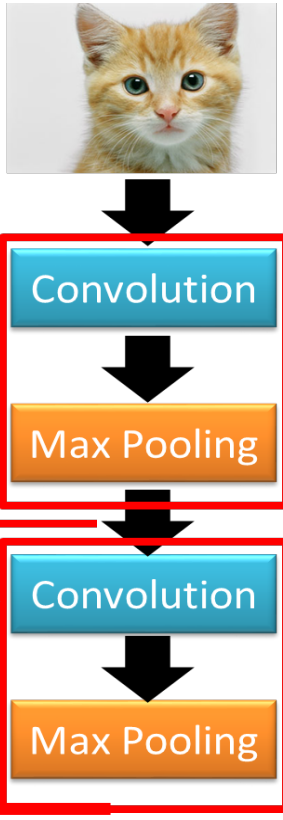
Question

- Will we get 25^2 feature map, if we use 25 filter to do two rounds of convolution?









Filter becomes a cubic, which height is 25.

No.

Training in CNN (1/2)

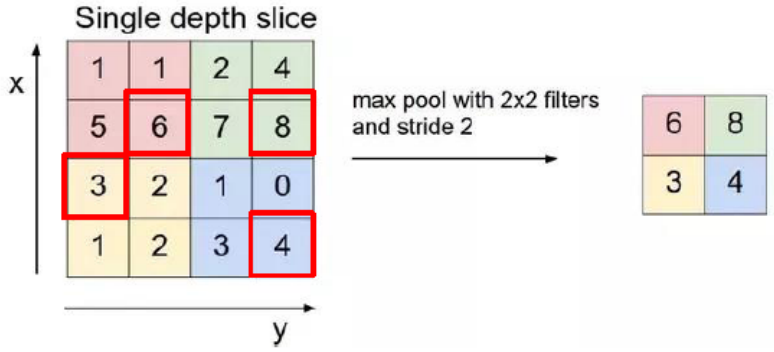
- Same as DNN
  - A lot of calculus

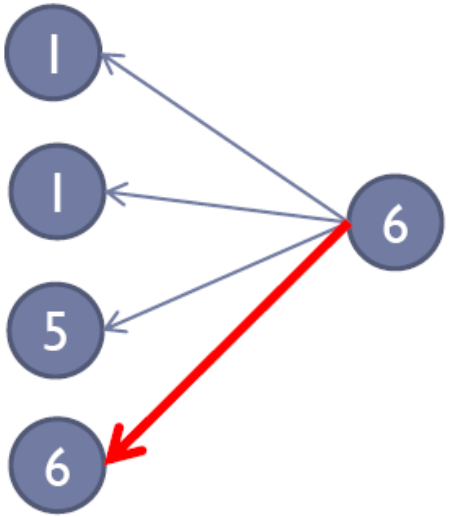
$$\partial C^r / \partial w_{ij}^l \quad ?$$

parameters are in conv layers and fully connected layers in CNN

Training in CNN (2/2)

- How to calculate max pooling derivative
  - Only backpropagate max value path
  - No gradient with respect to non maximum values (slightly change them does not affect the output)





Only backpropagate max value path.

Example-Max Pooling

The whole

cat dog ……

Convolution

Max Pooling

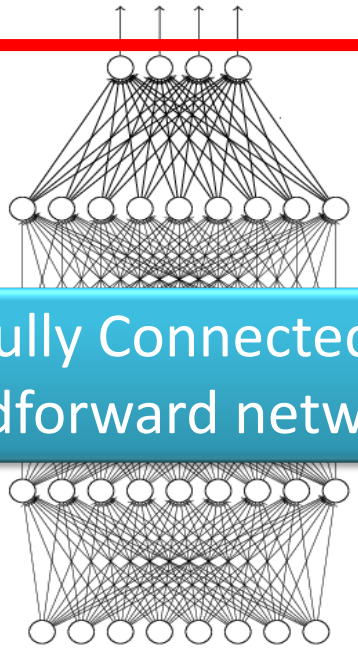
A new image

Convolution

Max Pooling

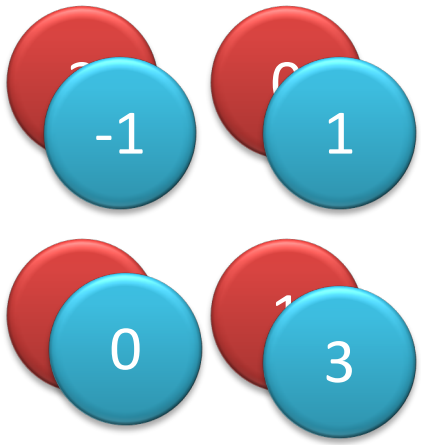
A new image

Flatten



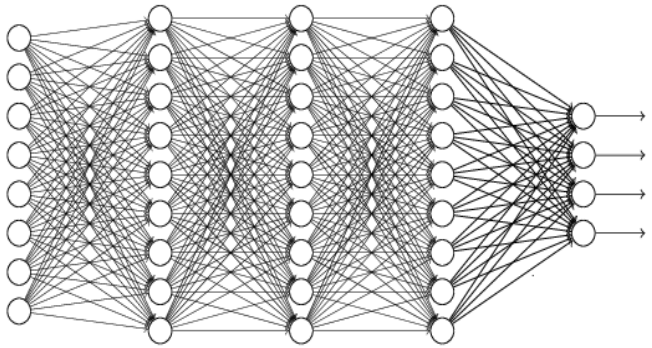
Fully Connected Feedforward network

Flatten



Flatten

3
0
1
3
-1
1
0
3

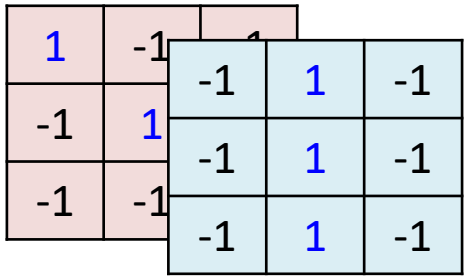


Fully Connected Feedforward network

CNN in Keras

Only modified the ***network structure*** and ***input format (vector -> 3-D tensor)***

```
model2.add( Convolution2D( 25,3,3,
            input_shape=(1,28,28) ) )
```



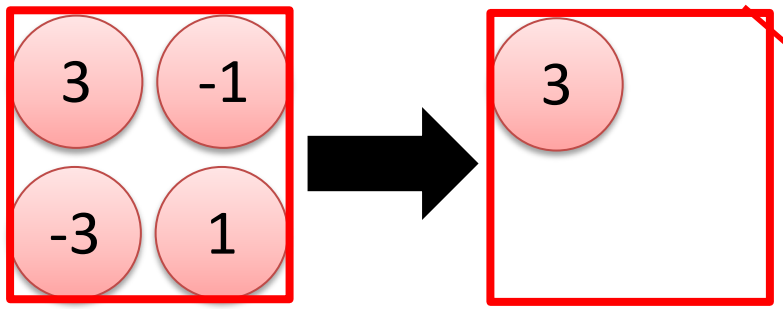
There are 25 3x3 filters.

Input_shape = (1 , 28 , 28)

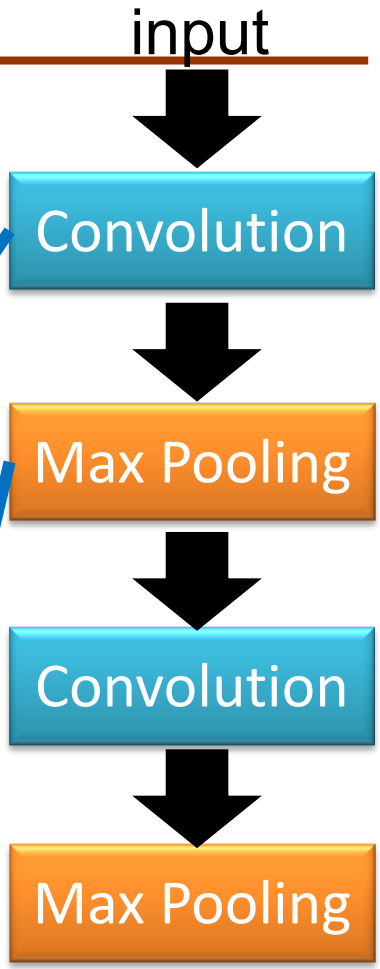
1: black/weight, 3: RGB

28 x 28 pixels

```
model2.add(MaxPooling2D((2,2)))
```



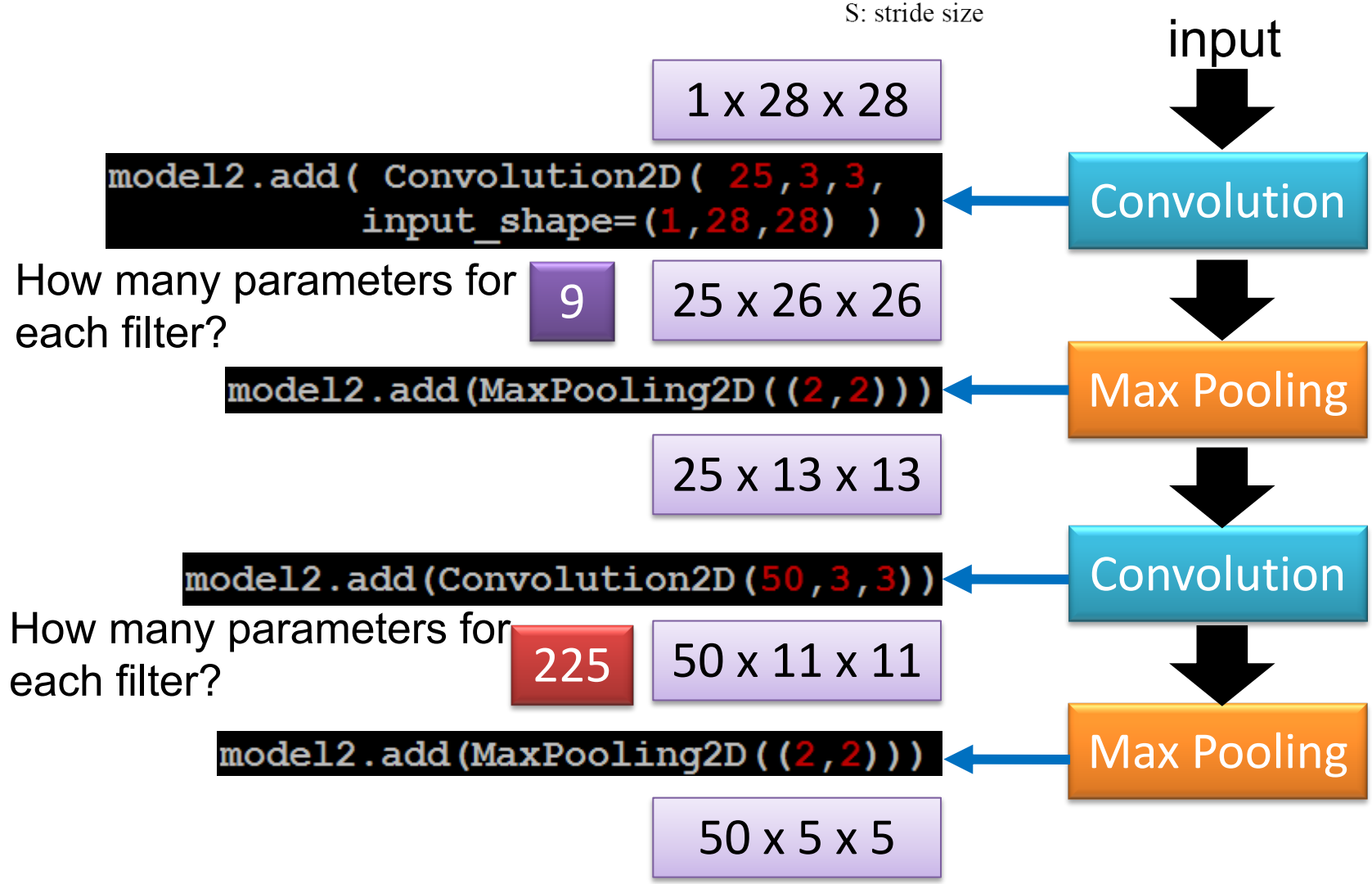
Get 2*2 to do Max Pooling.



CNN in Keras

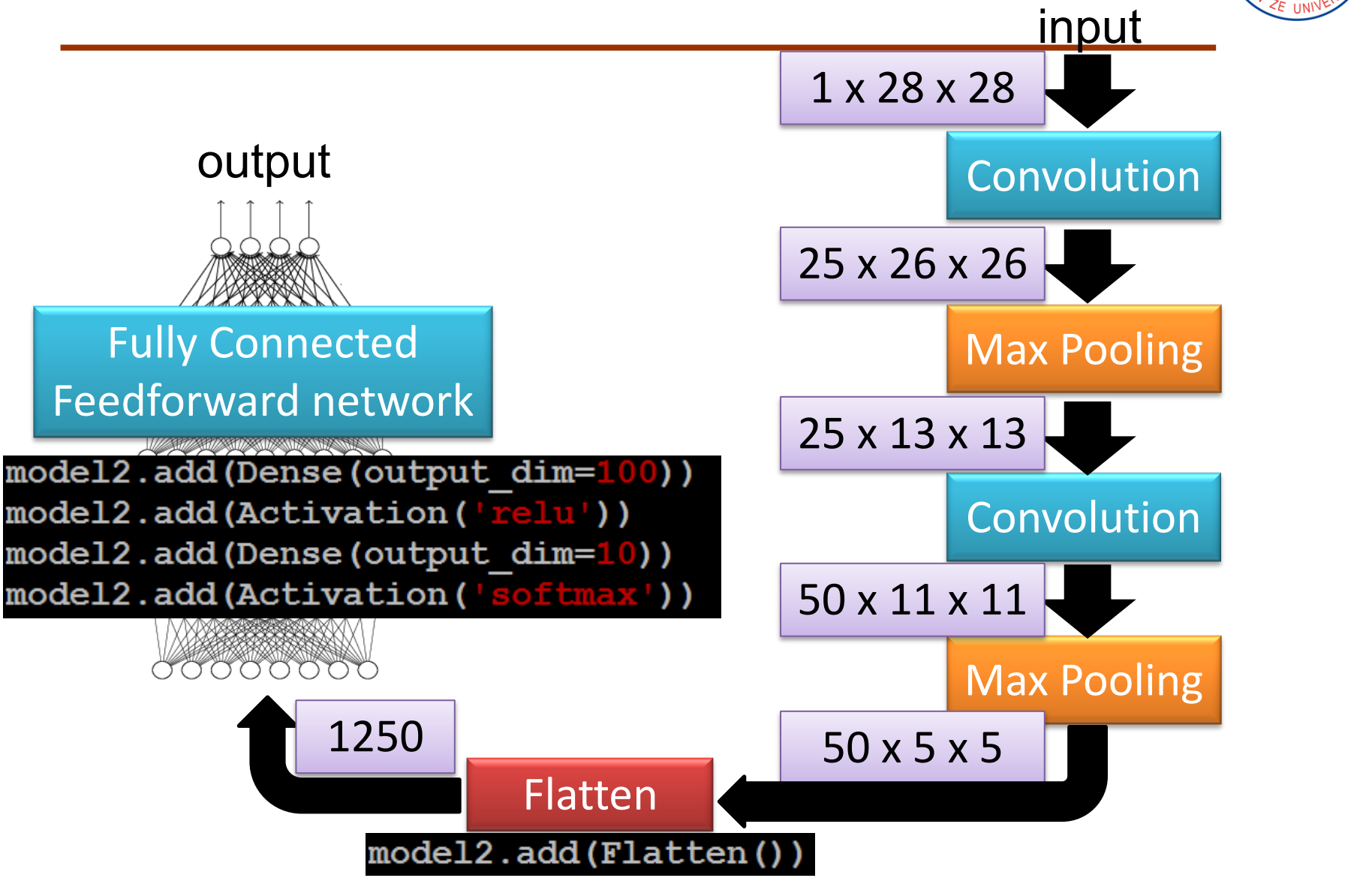
$$O = \frac{(W - K + 2P)}{S} + 1$$

O: output height/length
W: input height/length
K: filter size
P : padding
S: stride size



CNN in Keras

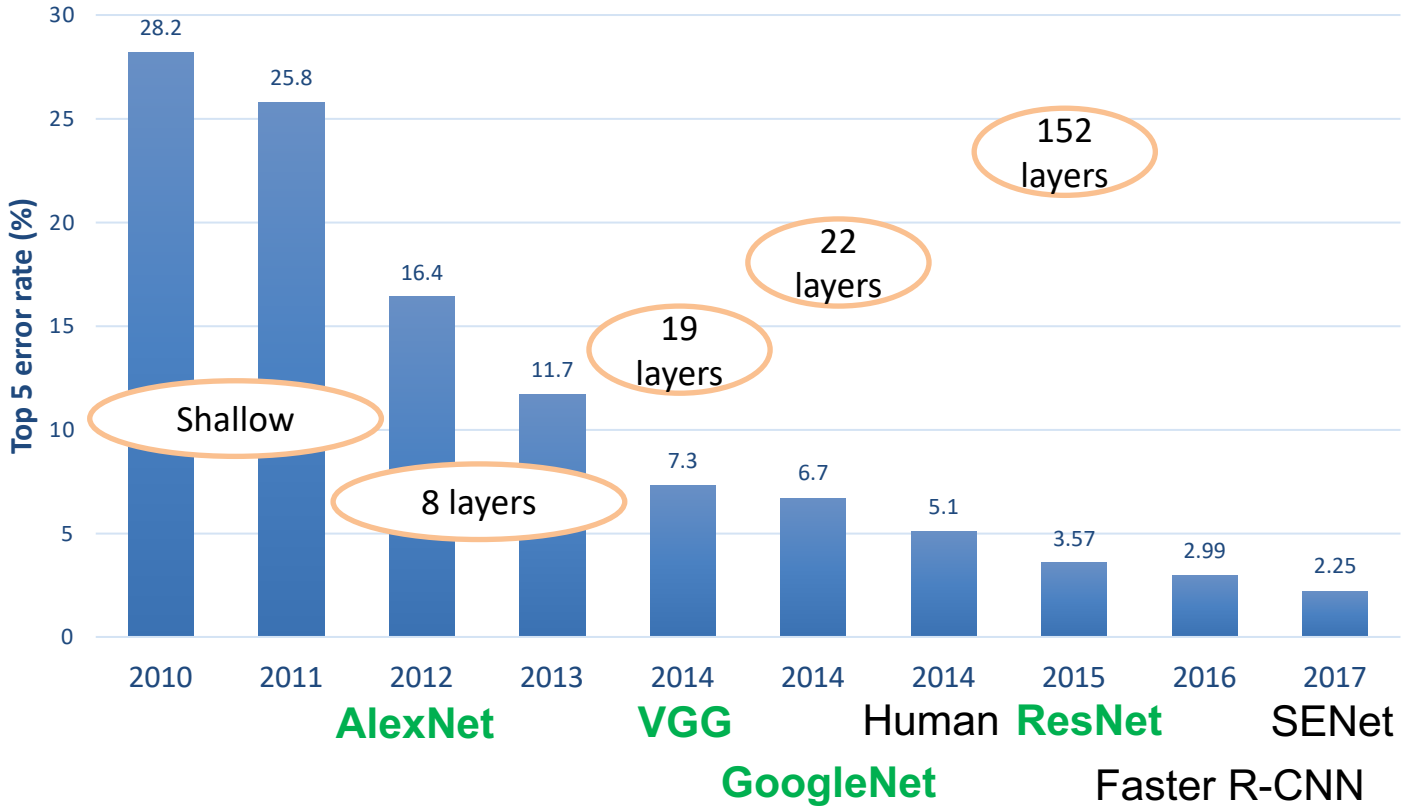
Only modified the ***network structure*** and ***input format (vector -> 3-D tensor)***



Famous CNN Model

ILSVRC 競賽 (1/2)

- ImageNet Large Scale Visual Recognition Competition (ILSVRC)
- 2010~2017



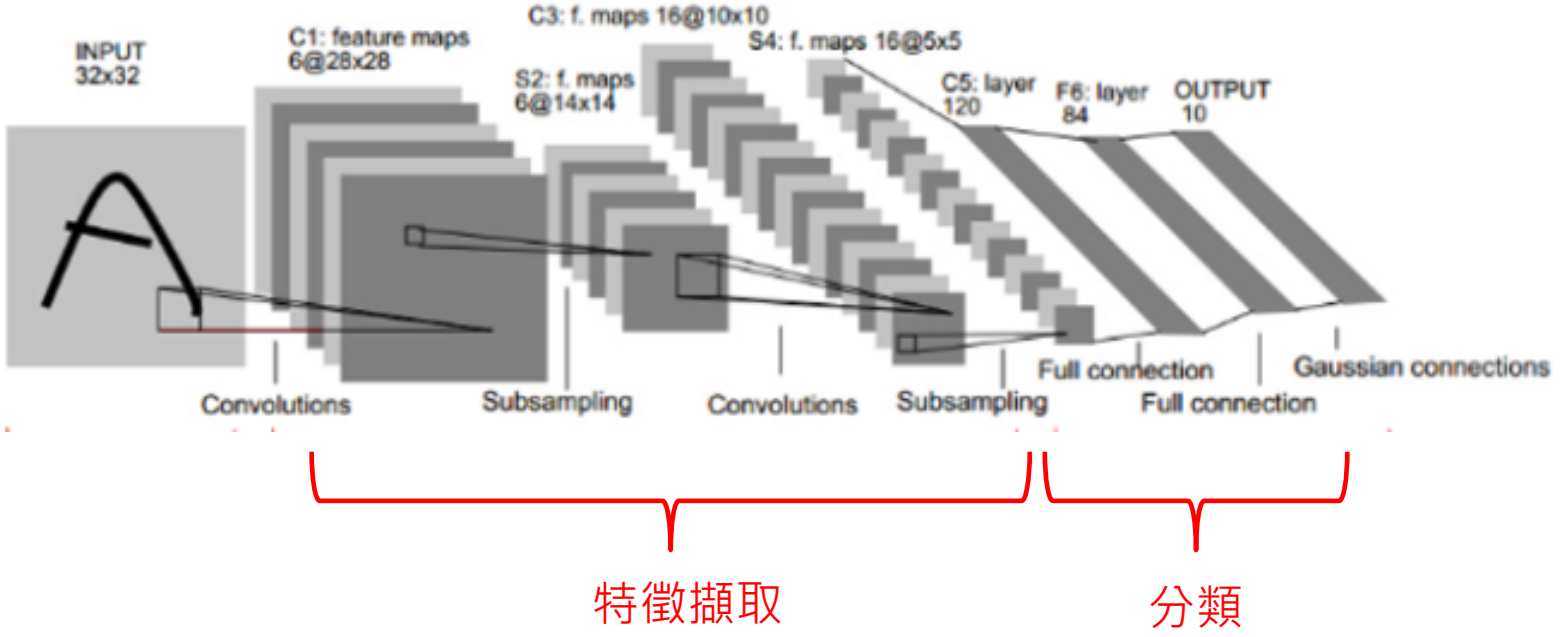
ILSVRC 競賽 (2/2)

- ImageNet
  - 是一個非常大的影像資料庫
  - 包含1500萬的影像以及22000種類別
  - 在比賽中，有120萬張影像有1000個標記(label)

- WebVision
  - 240萬張的影像，沒有任何的標記 (避免人為的錯誤)
  - 主題：
    - 物件偵測:偵測物件並且標記
    - 轉移式學習: 採用第一個模型轉移到其他資料集

LeNet

- CNN的父親。

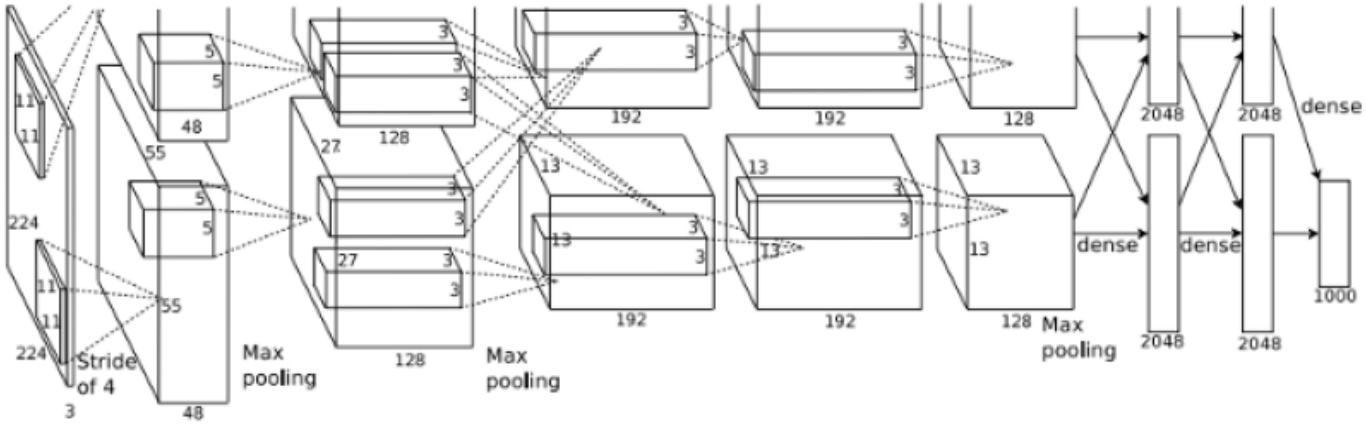


AlexNet

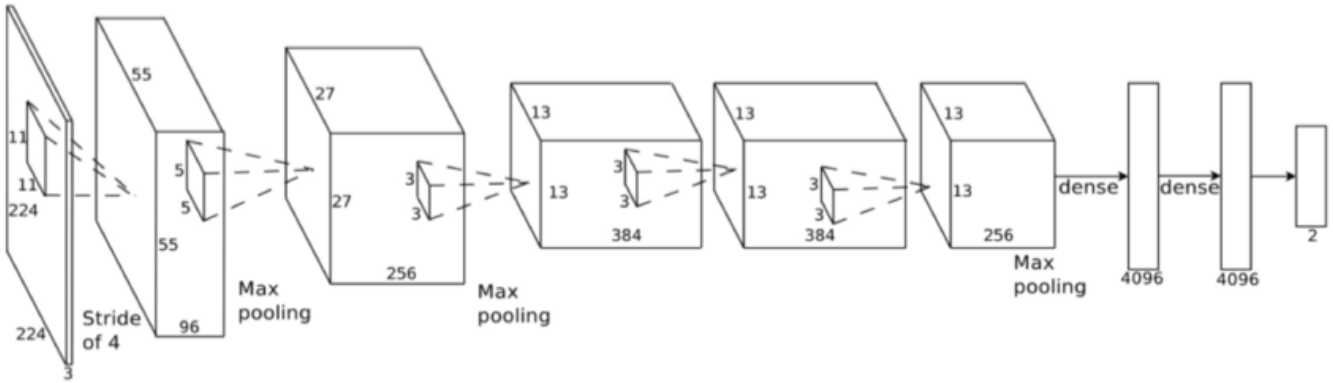
- 2012年， Prof. Hinton使用AlexNet 在2012年的ImageNet LSVRC的比賽上拿了冠軍(error rate 16.42%.)，而且準確率完勝第二名
- 那時候造成很大的轟動，也使得CNN開始被重視，變成所謂的CNN大時代
- 之後的比賽也都是由CNN拿下冠軍，深度學習正式大爆發。

AlexNet

- 由於運算資源通常是不夠的，
- 因此拆成兩個網路，用兩塊GPU去運算。

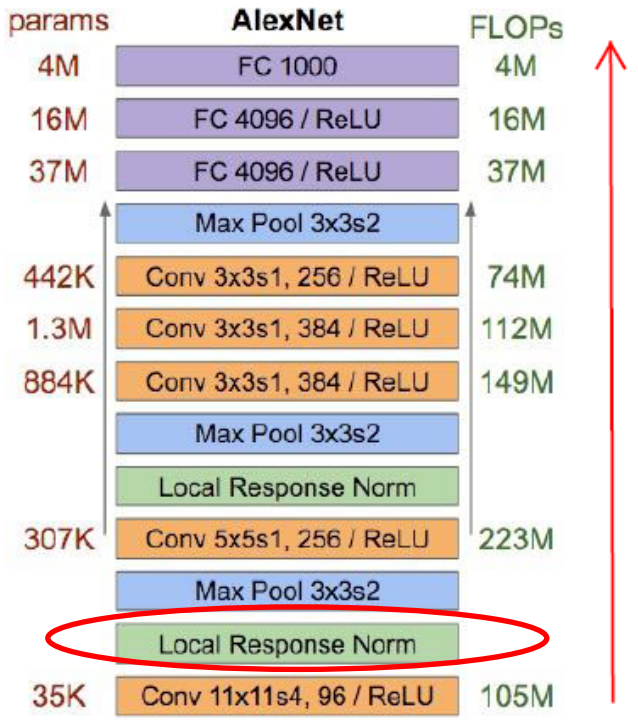


- 合併成一個網路的樣子



AlexNet

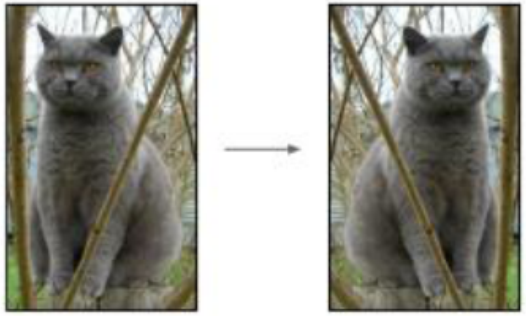
- 5個卷積層和3個全連接層
- LRN: Local Response Normalization
  - 跨feature map進行normalization，讓資料的差異性更大



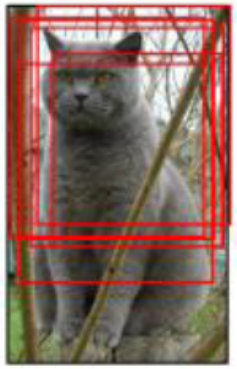
Use 11*11 filter to do convolution

fc: fully connected layer

在AlexNet中資料增量



Flip horizontally



Random crops/scales

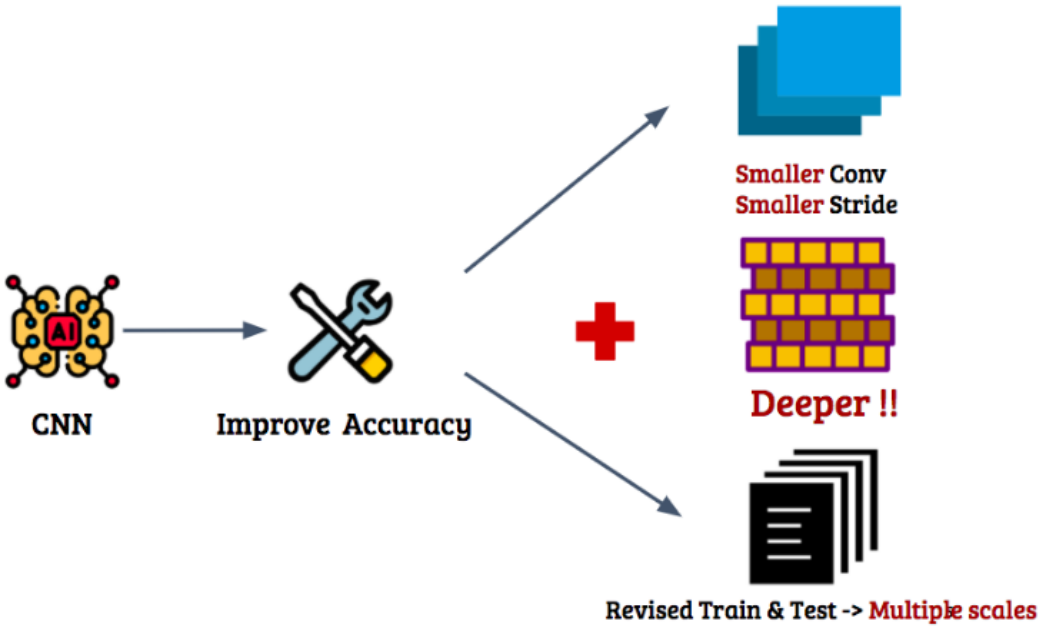
Color jittering

AlexNet 小結

- 使用ReLU 作為激活函數
  - 證明了比使用sigmoid效果好
- 提出dropout來避免overfitting
- 使用 max pooling，而不是 mean pooling
- 使用 Local Response Normalization (LRN) 建立區域性神經元的競爭機制，強化response較大的神經元，抑制response較小的神經元
- 使用兩張 GTX 580 來加速運算
- 在資料集中進行資料增量

VGG

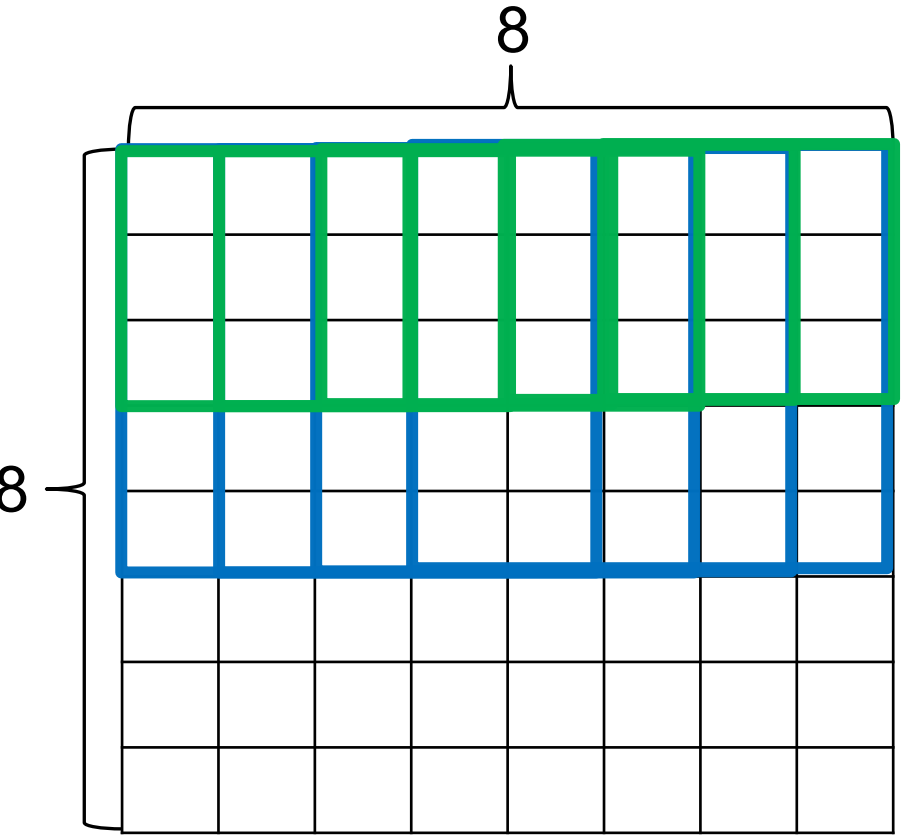
- 當時改善CNN的效能可以從兩個方向著手
  - 使用更小的Conv或是更小的stride
  - 利用不同的Data Augmentation，例如: Multiple Scale training等
- 而在VGG的paper中主要是將兩個合併在一起，提供一個更深且結果穩定的網路。



VGG原始paper: https://arxiv.org/pdf/1409.1556.pdf

Conv of VGG

- VGG最重要的概念就是使用大量的3 X 3 Conv
  - 作者認為將較大的Conv抽換成較小的Conv可以將information 量提高
- 使用較多的較小的Conv比大的Conv可以減少參數



參數數量呢?

5×5 Conv => output 4×4 image — 25

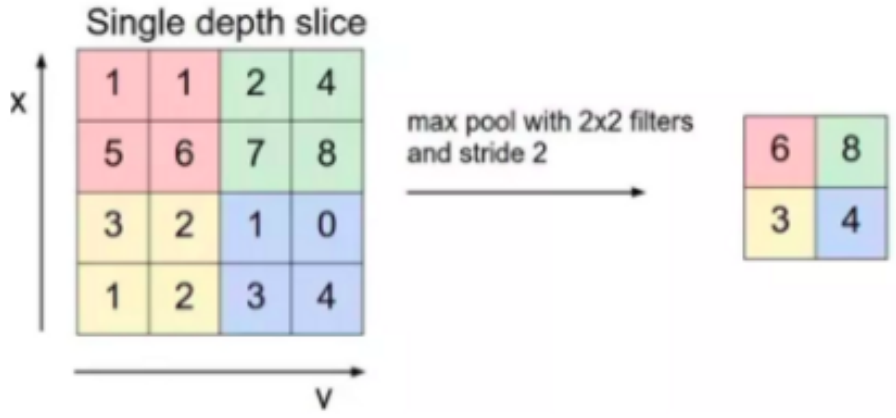
3×3 Conv => output 6×6 image
=>3×3 Conv => Output 4×4 image — 9+9

7×7 Conv => output 2×2 image — 49

3×3 Conv => output 6×6 image
=>3×3 Conv => Output 4×4 image
=>3×3 Conv => Output 2×2 image — 9+9+9

Pooling Layer of VGG

- AlextNet-> Pooling Layer = 3X3 and Stride =2 (Max Pooling)
- VGG-> Pooling Layer = 2X2 and Stride =2 (Max Pooling)
- 作者認為2X2 pooling可得到更多的資訊量，因此相較AlexNet 3X3 pooling，VGG改用更小的pooling，並不會overlap。

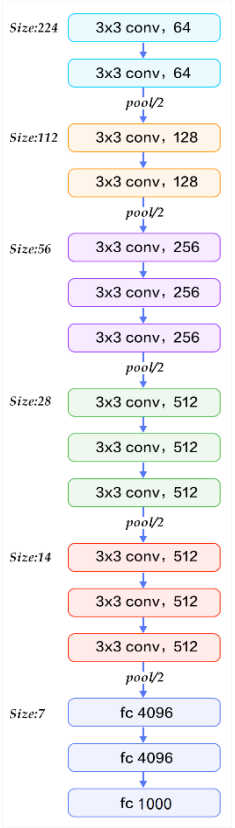


Multi-Scale Training / Multiple Crop Testing in VGG

- VGG在training以及testing資料上有做一些不一樣的處理
- Training的部分有使用Multiple scale training。
  - 在每次training 時，從一個固定的亂數範圍中，random一個數字，並縮放至那個數字，並隨機剪裁成所需大小。
- Testing使用多個crop進行預測
  - 將資料rescale成一個大小，並利用固定的crop大小預測左上、右上、左下、右下跟中間，並平均成最後預測結果。

VGG證明Deeper > shallow

- VGGNet 的深度比AlextNet更深
- 證明Deeper > shallow



用64個3x3的filter去做convolution

Pool: pooling, 2x2

fc: fully connected layer

Network	A,A-LRN	B	C	D	E
Number of parameters	133	133	134	138	144

VGGNet-16 VGGNet-19

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

ConvNet config. (Table 1)	smallest image side		top-1 val. error (%)	top-5 val. error (%)
	train (S)	test (Q)		
B	256	224,256,288	28.2	9.6
C	256	224,256,288	27.7	9.2
	384	352,384,416	27.8	9.2
	[256; 512]	256,384,512	26.3	8.2
D	256	224,256,288	26.6	8.6
	384	352,384,416	26.5	8.6
	[256; 512]	256,384,512	**24.8**	**7.5**
E	256	224,256,288	26.9	8.7
	384	352,384,416	26.7	8.6
	[256; 512]	256,384,512	**24.8**	**7.5**

VGG證明LRN無用

- A vs. A-LRN
- 從此A vs A-LRN實驗中，可得知
  加入LRN會降低準確率

Model Config	Image Size	Top1 error (%)	Top5 error (%)
A	256	29.6	10.4
A-LRN	256	29.7	10.5

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

*Top-1表示預測一次。
*Top-5表示預測五次，只要一次猜對就算正確。

VGG其他實驗: B vs C

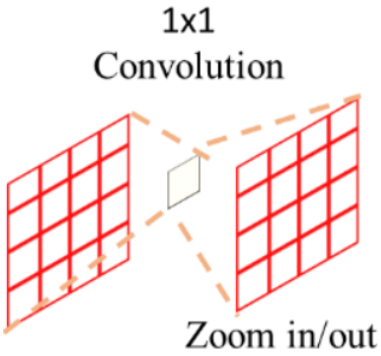
- 可發現若加入1 X 1 Conv，可提高準確度

Model Config	Image Size	Top1 error (%)	Top5 error (%)
B	256	28.7	9.9
C	256	28.1	9.4

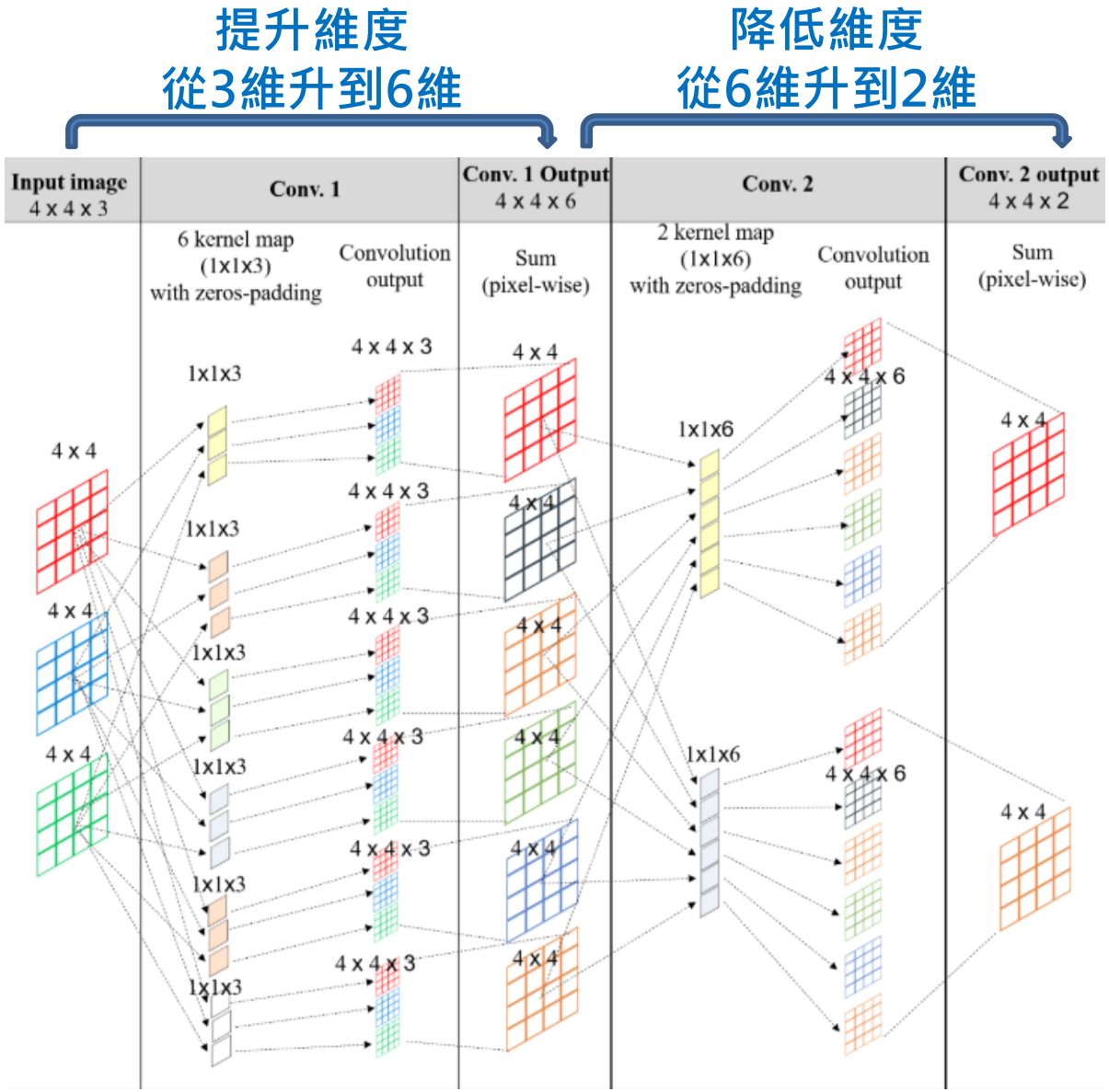
ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

1×1 Conv (1/3)

- 1×1 Conv
  - 把輸入的圖裡面的值做放大或是縮小
  - 感覺沒什麼用，的確沒什麼用，但1×1捲積真的用途重點不是在作卷積這件事情。
- 1×1的Conv可以拿來做降維或是升維



1×1 Conv (2/3)


提升維度
從3維升到6維
降低維度
從6維升到2維
Input image
4 x 4 x 3
Conv. 1
Conv. 1 Output
4 x 4 x 6
Conv. 2
Conv. 2 output
4 x 4 x 2
6 kernel map
(1x1x3)
with zeros-padding
Convolution
output
Sum
(pixel-wise)
2 kernel map
(1x1x6)
with zeros-padding
Convolution
output
Sum
(pixel-wise)
4 x 4
1x1x3
4 x 4 x 3
1x1x6
4 x 4 x 6

1×1 Conv (3/3)

- 1×1 Con還可以減少訓練參數
- input的大小為100x100x56
- 如果我們做一個有125個filter的5x5卷積層之後(stride = 1 , pad = 2)，輸出將為 96x96x125
- 參數: 56*5*5*125 = 175000

- 如果我們先經過28個filter的1x1的卷積，再經過有125個filter的5x5卷積層之後，輸出一樣為96x96x125
- 參數: 56*1*1*28+28*5*5*125=89068

- 相對少了快一半的參數

VGG其他實驗:C vs D

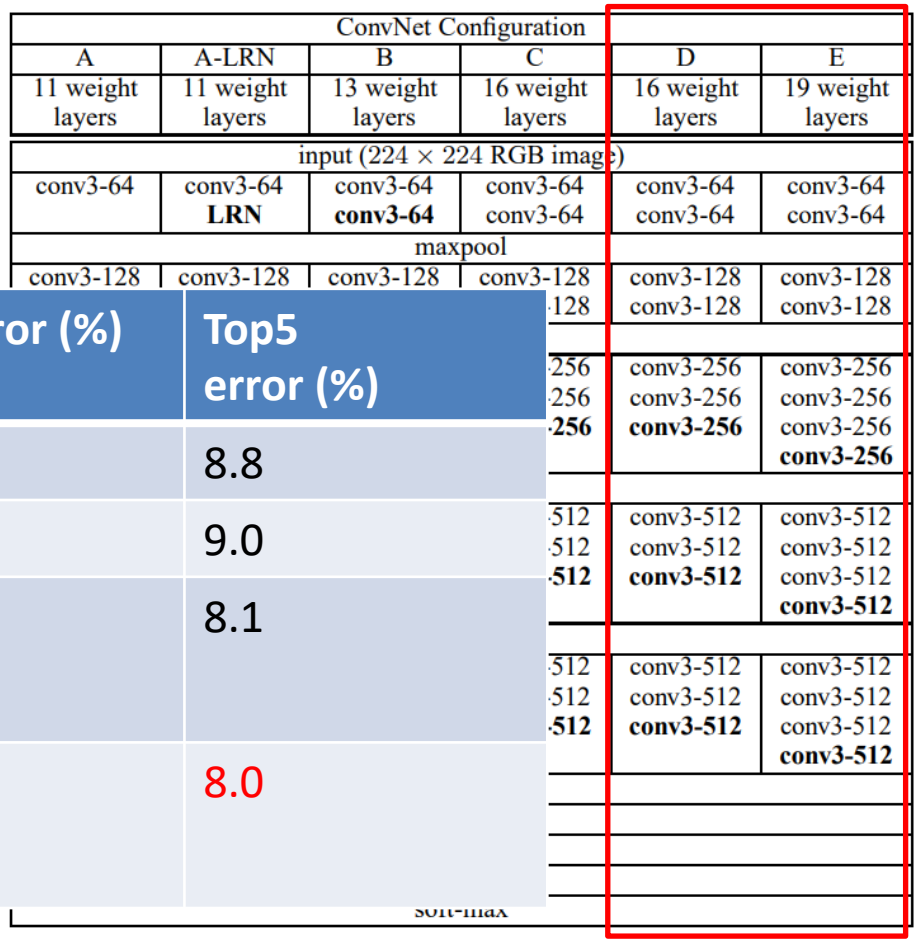
- C vs D
- 將1 X 1 Conv替換成 3 X 3 Conv可發現結果也更好了
  - 作者解釋為3 X 3 Conv相較於 1 X 1 Conv可得到更多的空間資訊量。

Model Config	Image Size	Top1 error (%)	Top5 error (%)
C	256	28.1	9.4
D	256	27.0	8.8

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

VGG其他實驗: D vs E (1/2)

Model Config	Image Size	Top1 error (%)	Top5 error (%)
D	256	27.0	8.8
E	256	27.3	9.0
D	Train[256;512] Test[384]	25.6	8.1
E	Train[256;512] Test[384]	25.5	8.0



VGG其他實驗: D vs E (2/2)

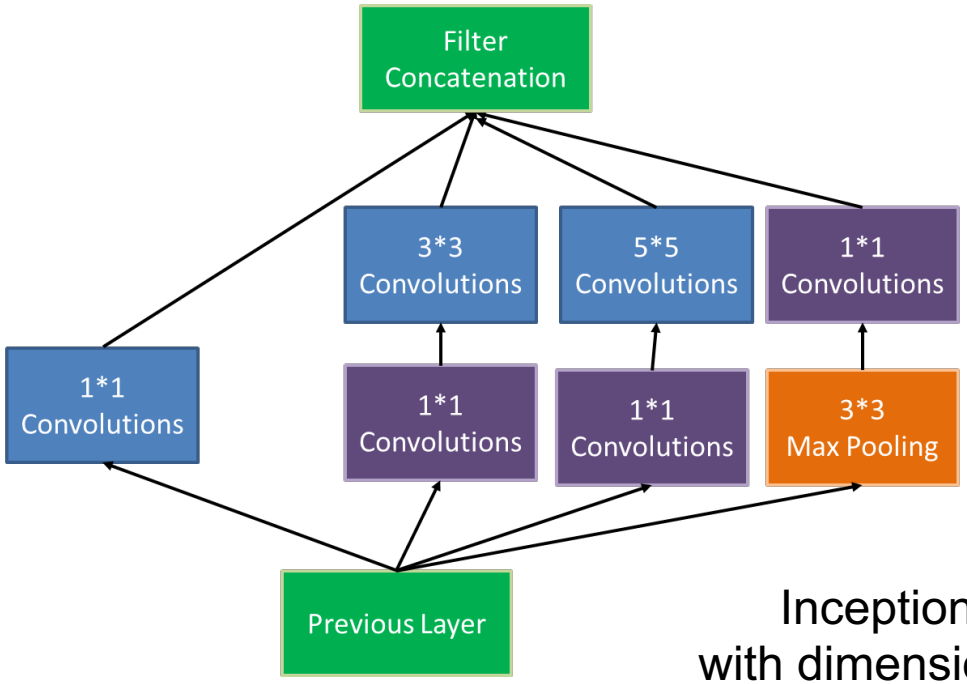
- 在D跟E的實驗比較中，發現在這樣的深度中，效果不一定越深越好
- 需要做更多的Data Augmentaion，才可以讓更多更深的模型學得更好
- 這是VGG模型的缺陷，當模型越來越深，資料在每一層的耗損越來越高(Ex: **Gradient Vanishing**)，使得在後面幾層就無法學習出好的結果。
- 這個問題在ResNet有被解決，因此RestNet才可以疊出幾百層。

VGGNet 小結

- 可得到較好的準確度
  - 使用較小的Conv
  - 使model變深
- LRN對模型較無太大的幫助。

GoogleNet (1/2)

- Inception 模組: 在GoogleNet中被提出的新架構，因此GoogleNet又被稱為 “Inception Network” 。
- Filter size往往不知道要設定多少，因此GoogleNet在網路中結合不同的filter來解決filter設定的問題。
- 先通過1x1 Conv來達到降維和減少參數。



Inception module with dimension reduction.

GoogleNet (2/2)

Conv	Size=3*3 /stride= 2	299*299*3
Conv	Size=3*3 /stride= 1	149*149*32
Conv	Size=3*3 /stride= 1	147*147*32
Pooling	Size=3*3 /stride= 2	147*147*64
Conv	Size=3*3 /stride= 1	73*73*64
Conv	Size=3*3 /stride= 2	71*71*80
Conv	Size=3*3 /stride= 1	35*35*192
Inception module	3*inception module	35*35*288
Inception module	5*inception module	17*17*768
Inception module	3*inception module	8*8*1280
Pooling	8*8	8*8*2048
Linear	Logits	1*1*2048
Softmax	Classification	1*1*1000

直接連接softmax。

GoogleNet 小結

- 可以更好的控制參數
  - 移除了全連接網路，直接連結softmax。
  - 參數數量是AlexNet的一半。

ResNet (1/2)

- 退化問題 (Degradation problem)
  - 越多層，反而錯誤率越高。

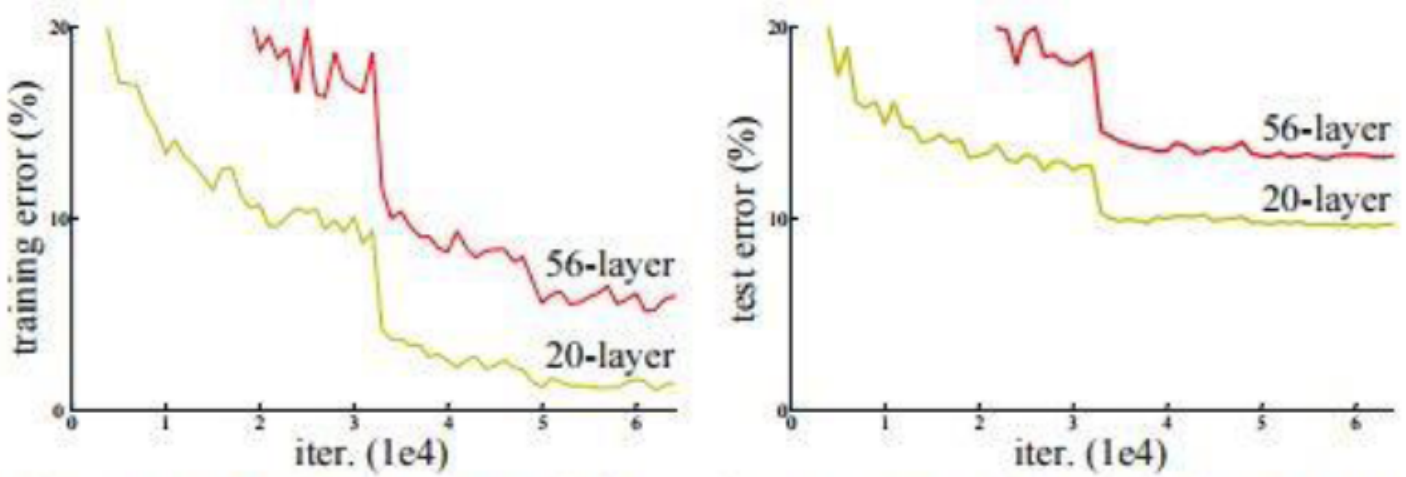
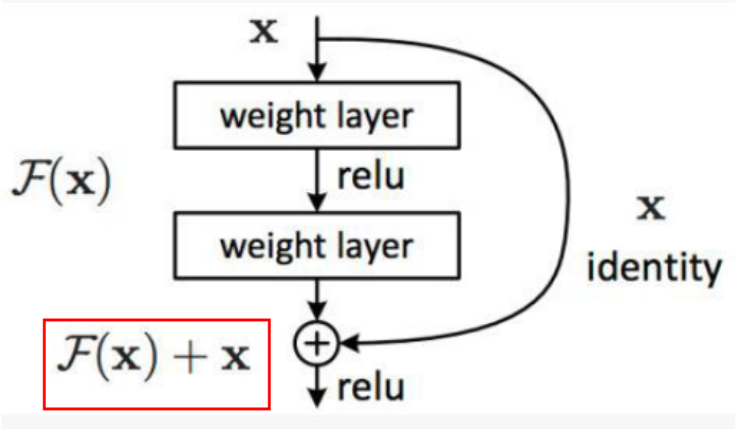
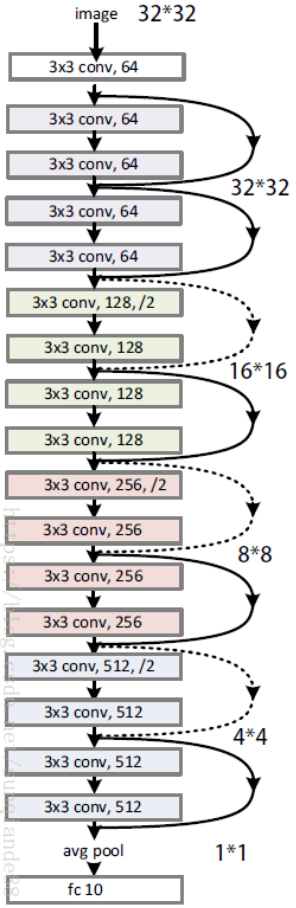


Figure 1. Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer "plain" networks. The deeper network has higher training error, and thus test error. Similar phenomena on ImageNet is presented in Fig. 4.

ResNet (2/2)

- 越多層 => “以訛傳訛” => 如果傳的人越多，錯誤率越高
- 越多層，也會更容易造成梯度消失的問題，ResNet加入捷徑來解決這個問題 。





ResNet 小結

- 解決退化議題
  - 使得深度學習網路真正可行。

What does CNN learn

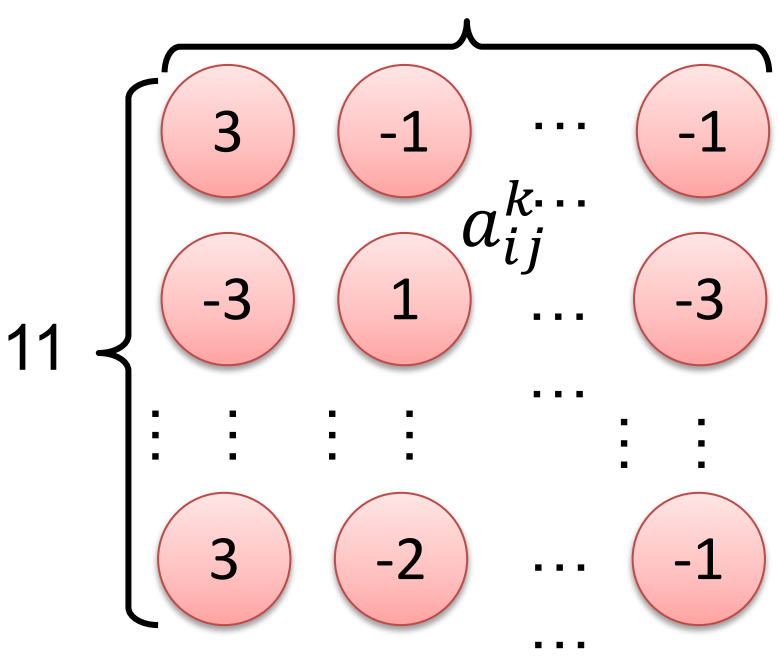
What does CNN learn?

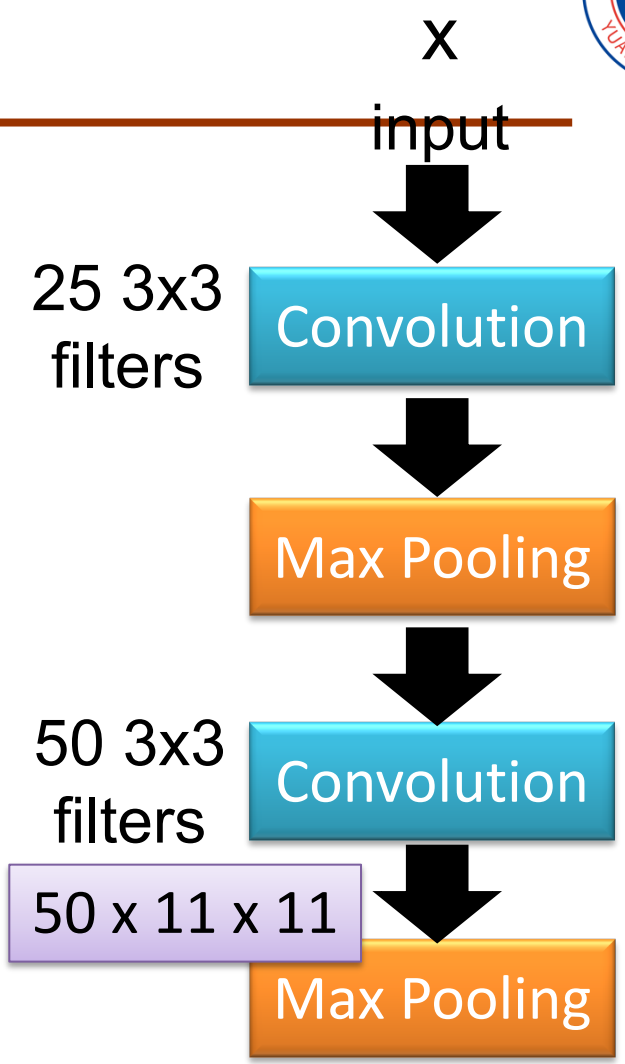
The output of the k-th filter is a 11 x 11 matrix.

Degree of the activation of the k-th filter:

$$a^k = \sum_{i=1}^{11} \sum_{j=1}^{11} a_{ij}^k$$

$$x^* = arg \max_x a^k \quad \text{(gradient ascent)}$$





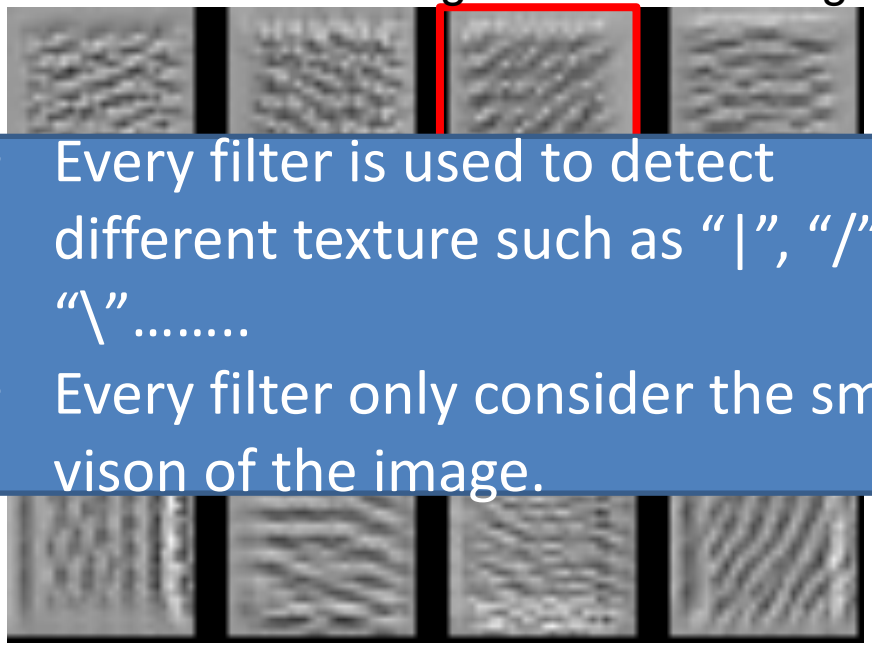
What does CNN learn?

The output of the k-th filter is a 11 x 11 matrix.

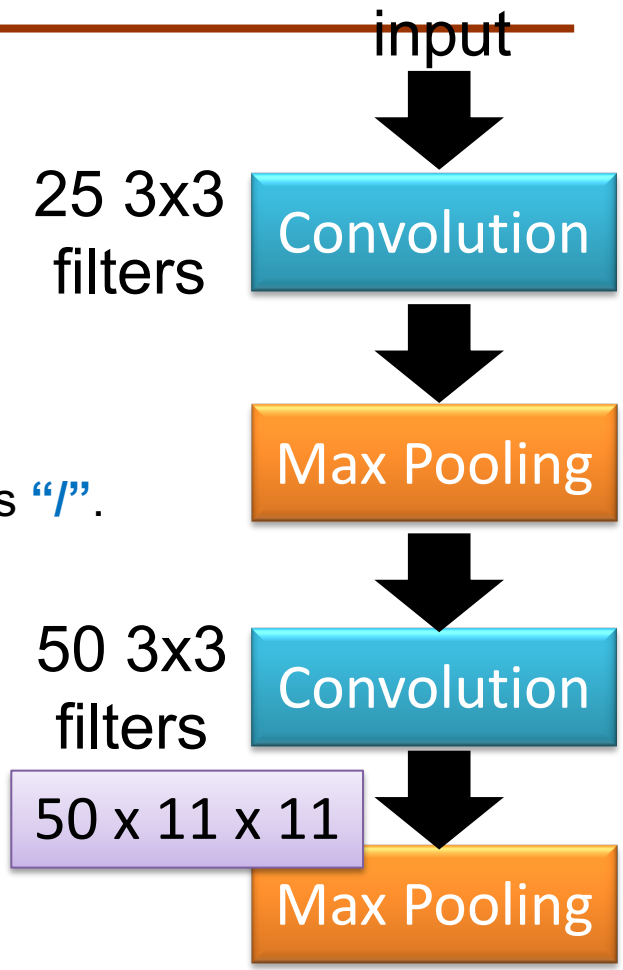
Degree of the activation of the k-th filter: $a^k = \sum_{i=1}^{11} \sum_{j=1}^{11} a_{ij}^k$

$x^* = arg \max_x a^k$ (gradient ascent)

This image reflects the filter is detecting whether the image contains "/".



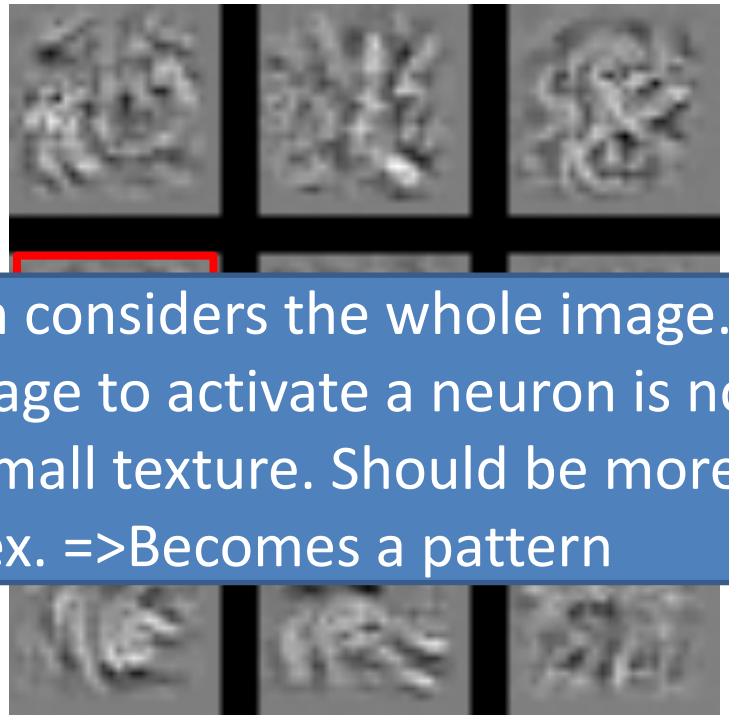
- Every filter is used to detect different texture such as "|", "/", "\".....
- Every filter only consider the small vison of the image.



For each filter

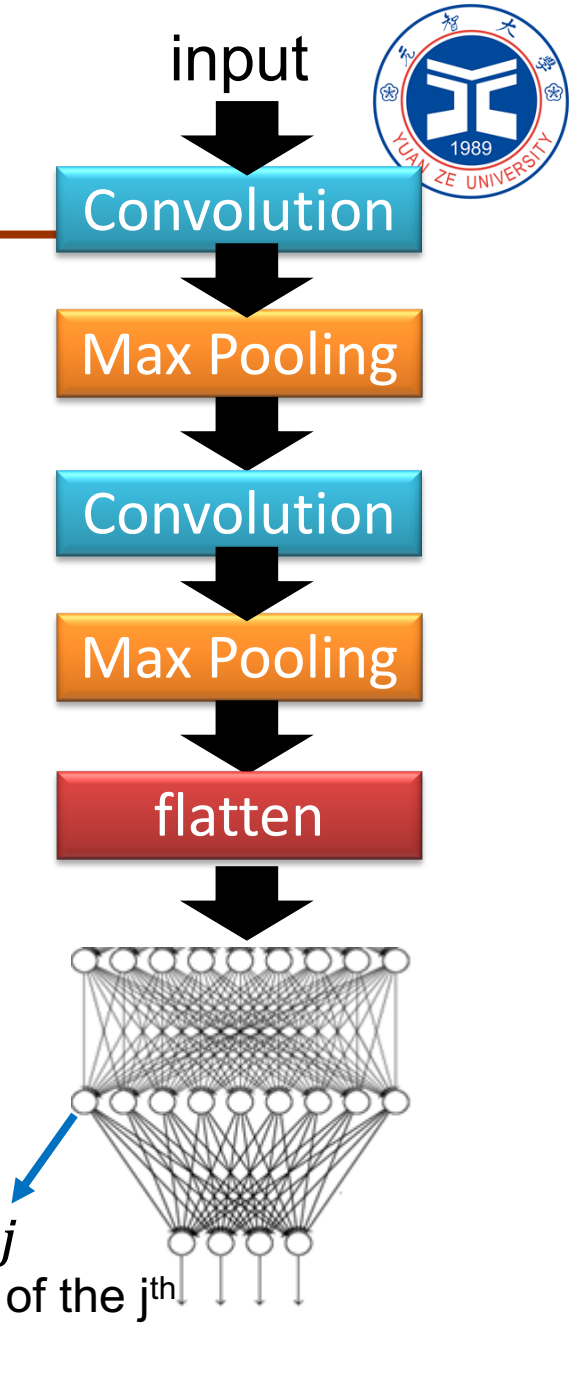
What does CNN learn?

Find an image maximizing the output of neuron: $x^* = arg\max_x a^j$



- Neuron considers the whole image.
- The image to activate a neuron is not like a small texture. Should be more complex. =>Becomes a pattern

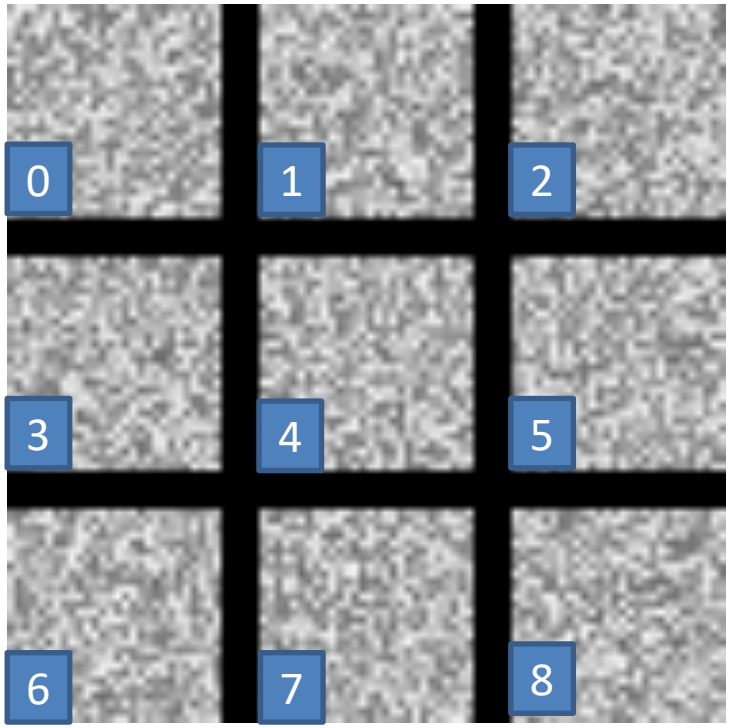
Each figure corresponds to a neuron

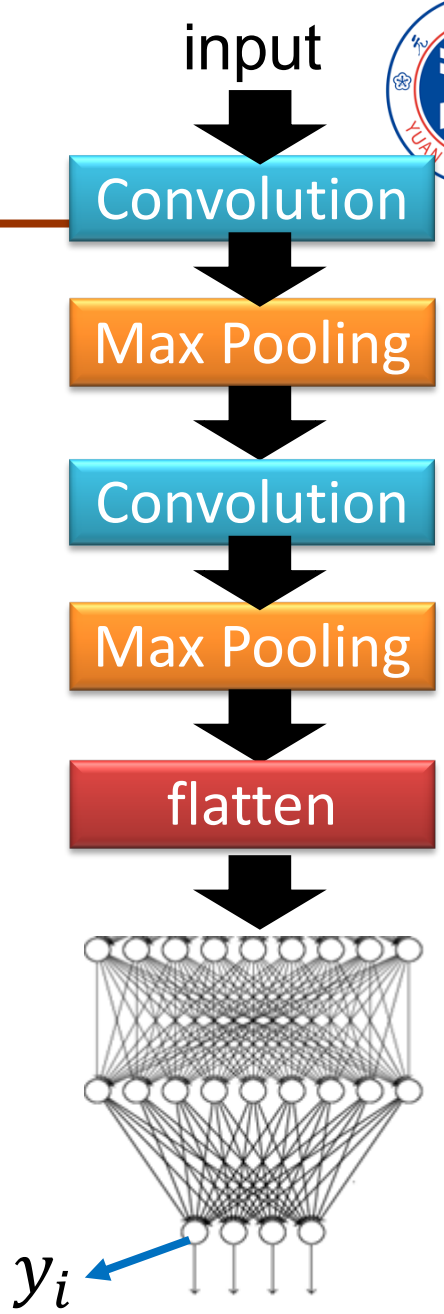


What does CNN learn?

$$x^* = arg\max_x y^i$$

Can we see digits?





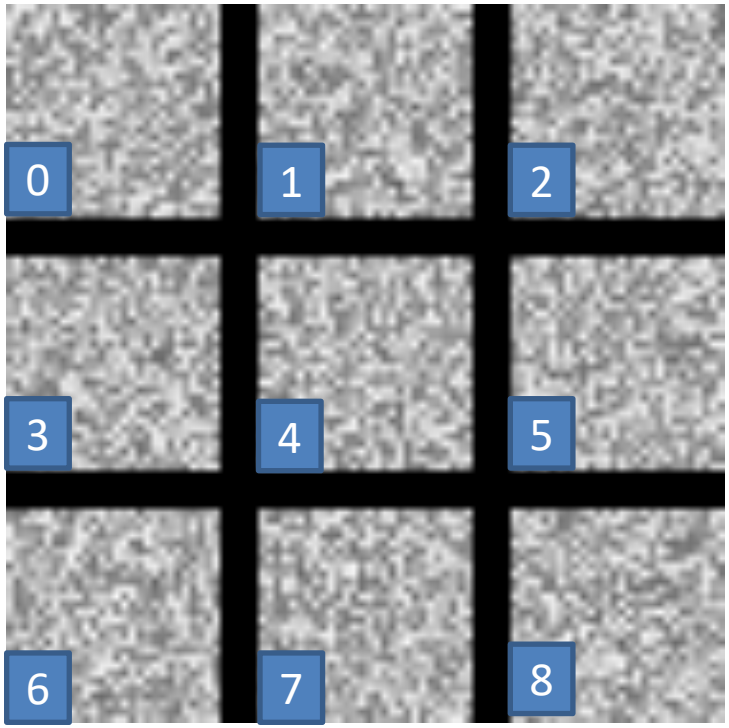
Deep Neural Networks are Easily Fooled
https://www.youtube.com/watch?v=M2IebCN9Ht4

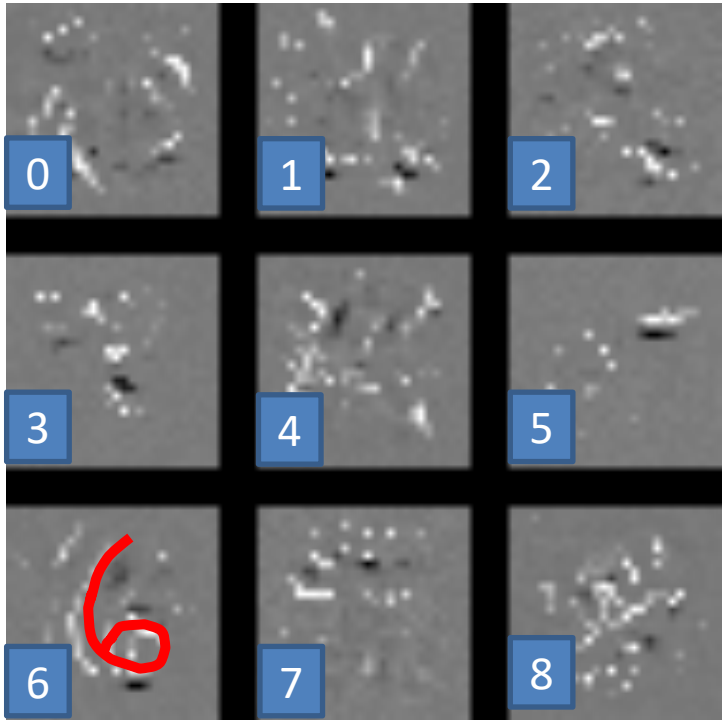
What does CNN learn?

Over all pixel values => L1 Regularization

$$x^* = arg\max_x y^i$$

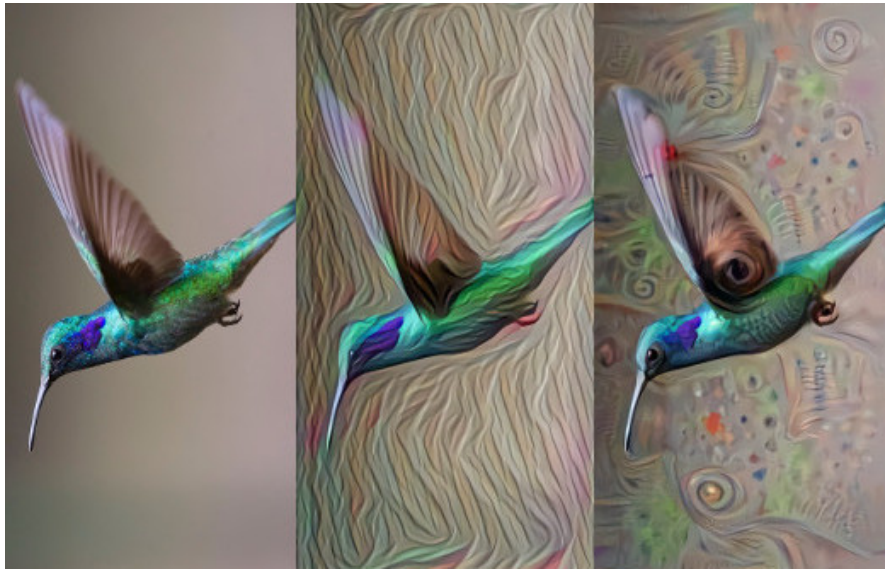
$$x^* = arg\max_x \left(y^i - \sum_{i,j} |x_{ij}| \right)$$





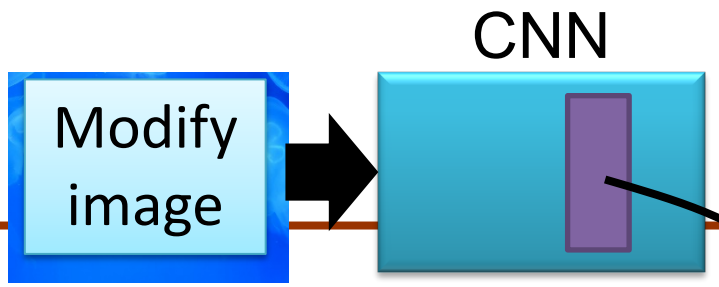
Deepdream

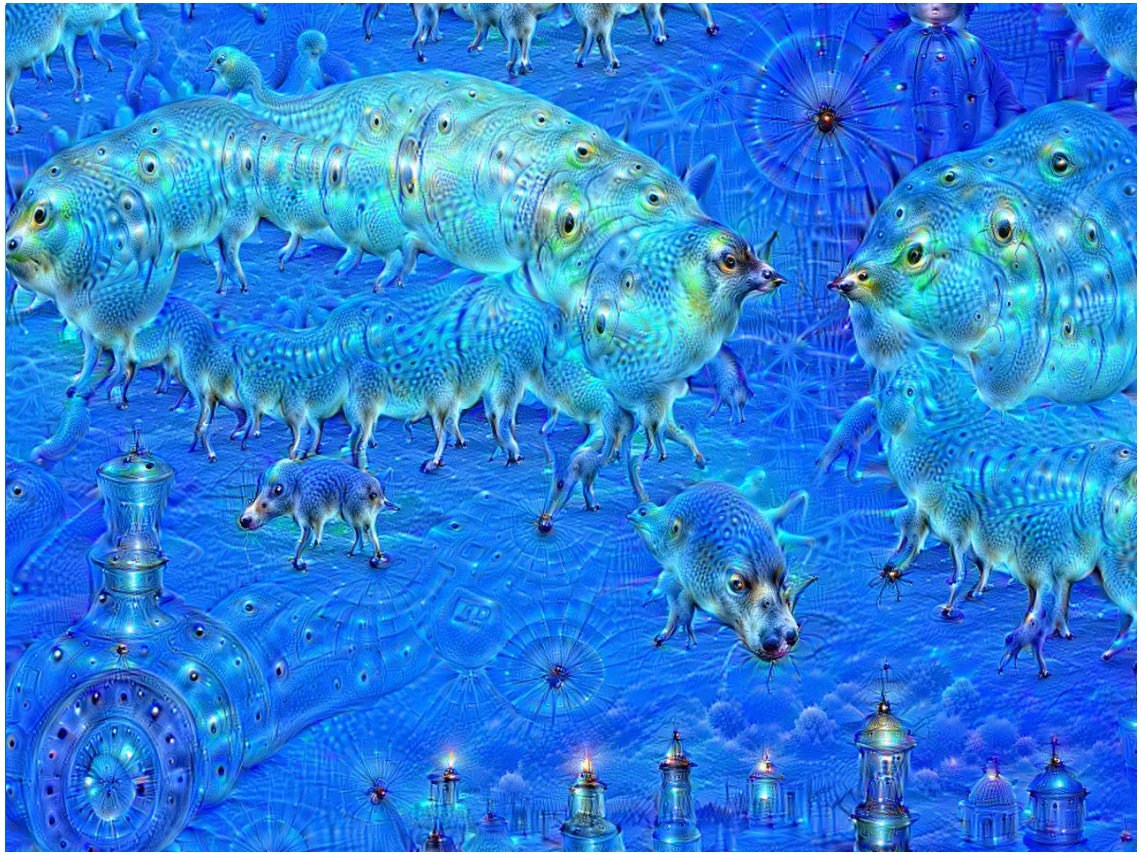
- Initially it was invented to help scientists and engineers to see what a deep neural network is seeing when it is looking in a given image.
- Later the algorithm has become a new form of psychedelic and abstract art.



https://deepdreamgenerator.com/

Deepdream





A hidden layer

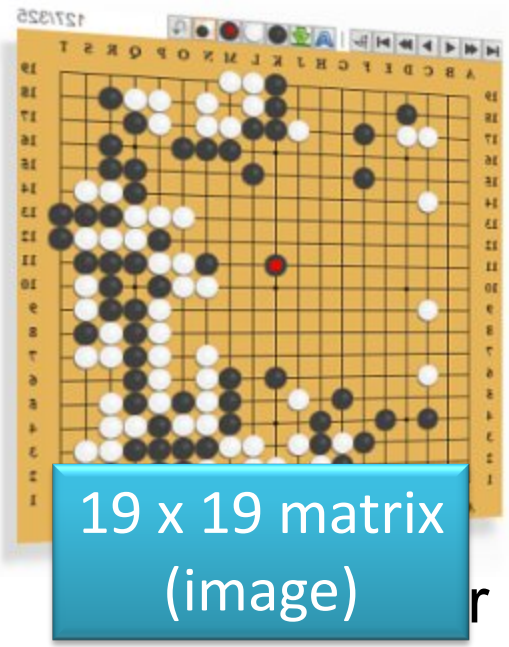
$$\begin{bmatrix} 3.9 \\ -1.5 \\ 2.3 \\ \vdots \end{bmatrix}$$

- Increase positive dimension.
- Decrease negative dimension.
- =>To emphasize what CNN has seen.

Deepdream Example

CNN Application

More Application: Playing Go



19 x 19 matrix (image)

Network

Next move (19 x 19 positions)

19 x 19 vector

Black: 1

white: -1

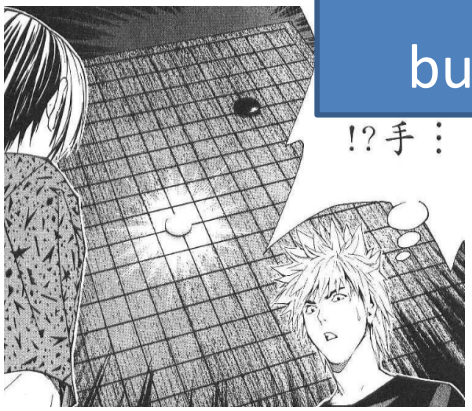
none: 0

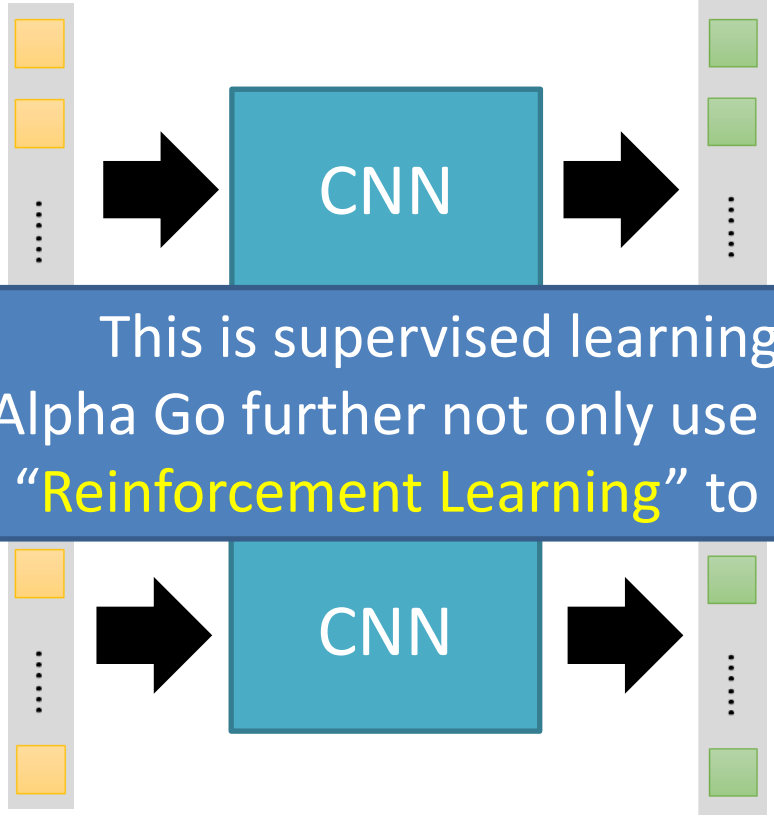
Fully-connected feedforward network can be used

But CNN performs much better.

More Application: Playing Go

Training: record of previous plays 黑: 5之五 → 白: 天元 → 黑: 五之5 …





Target:
“天元” = 1
else = 0

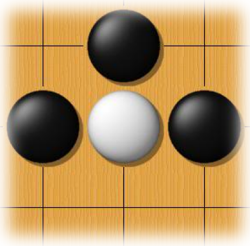
Target:
“五之 5” = 1
else = 0

This is supervised learning.
Alpha Go further not only use CNN,
but “Reinforcement Learning” to play Go.

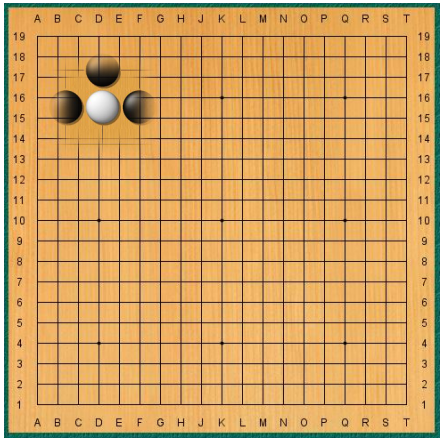
Why CNN for playing Go?

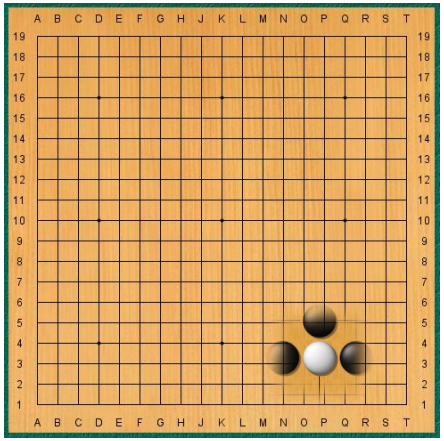
- Some patterns are much smaller than the whole image

Alpha Go uses 5 x 5 filter for first layer



- The same patterns appear in different regions.





Why CNN for playing Go?

- Subsampling the pixels will not change the object

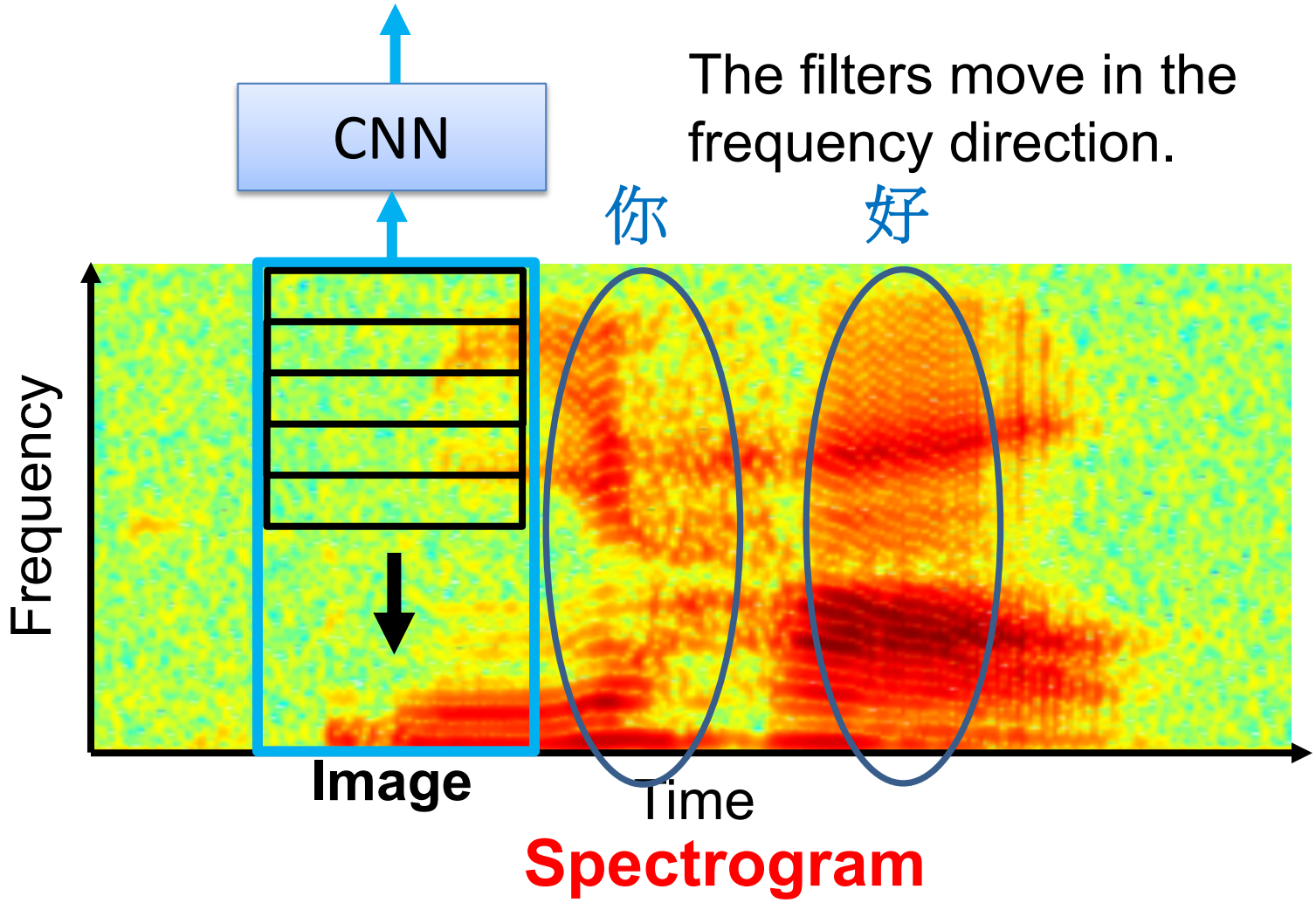
Max Pooling

How to explain this???

Neural network architecture. The input to the policy network is a 19 × 19 × 48 image stack consisting of 48 feature planes. The first hidden layer zero pads the input into a 23 × 23 image, then convolves *k* filters of kernel size 5 × 5 with stride 1 with the input image and applies a rectifier nonlinearity. Each of the subsequent hidden layers 2 to 12 zero pads the respective previous hidden layer into a 21 × 21 image, then convolves *k* filters of kernel size 3 × 3 with stride 1, again followed by a rectifier nonlinearity. The final layer convolves 1 filter of kernel size 1 × 1 with stride 1 with a different bias for each position, and applies a softmax function. The Extended Data Table 3 additionally show the results of training with $k = 128$, 256 and 384 filters.

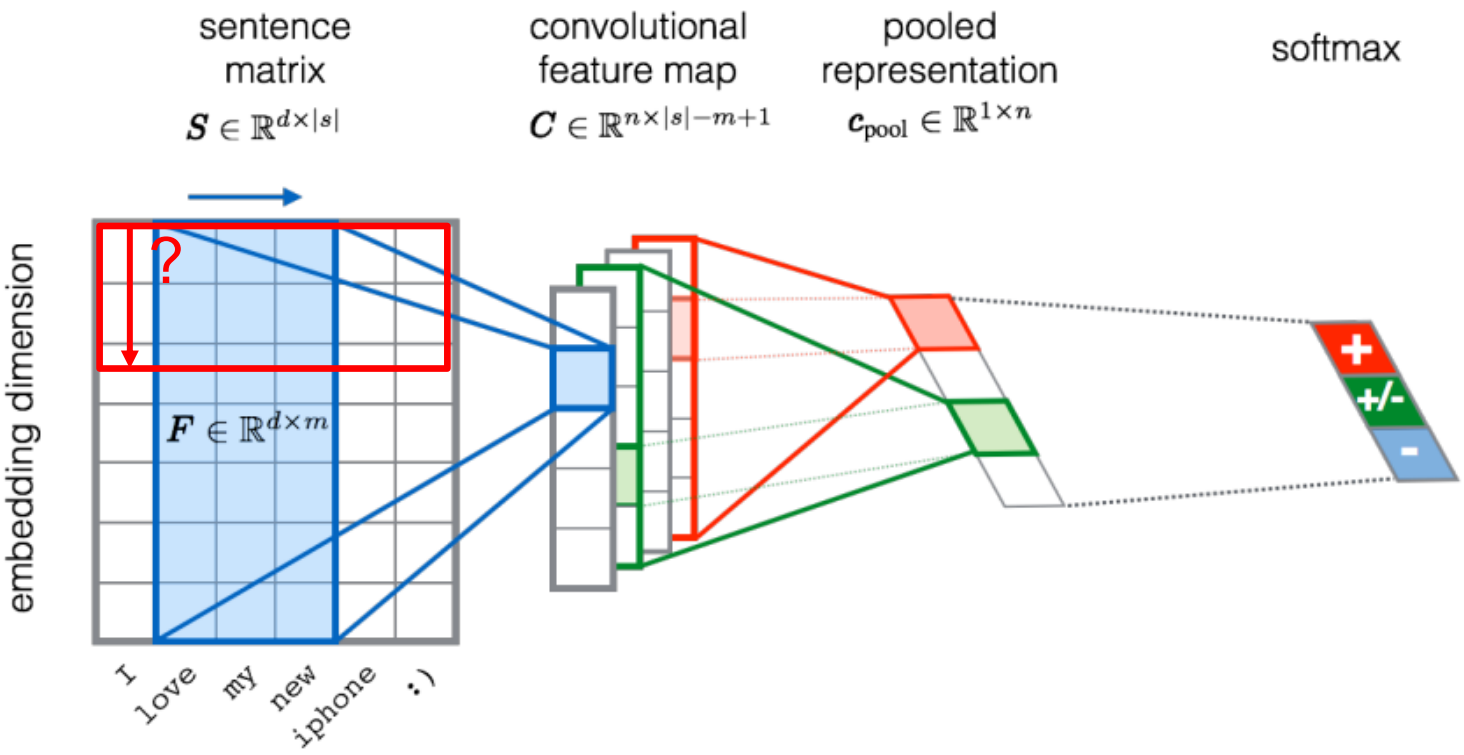
Alpha Go does not use Max Pooling .....

More Application: Speech



More Application: Text

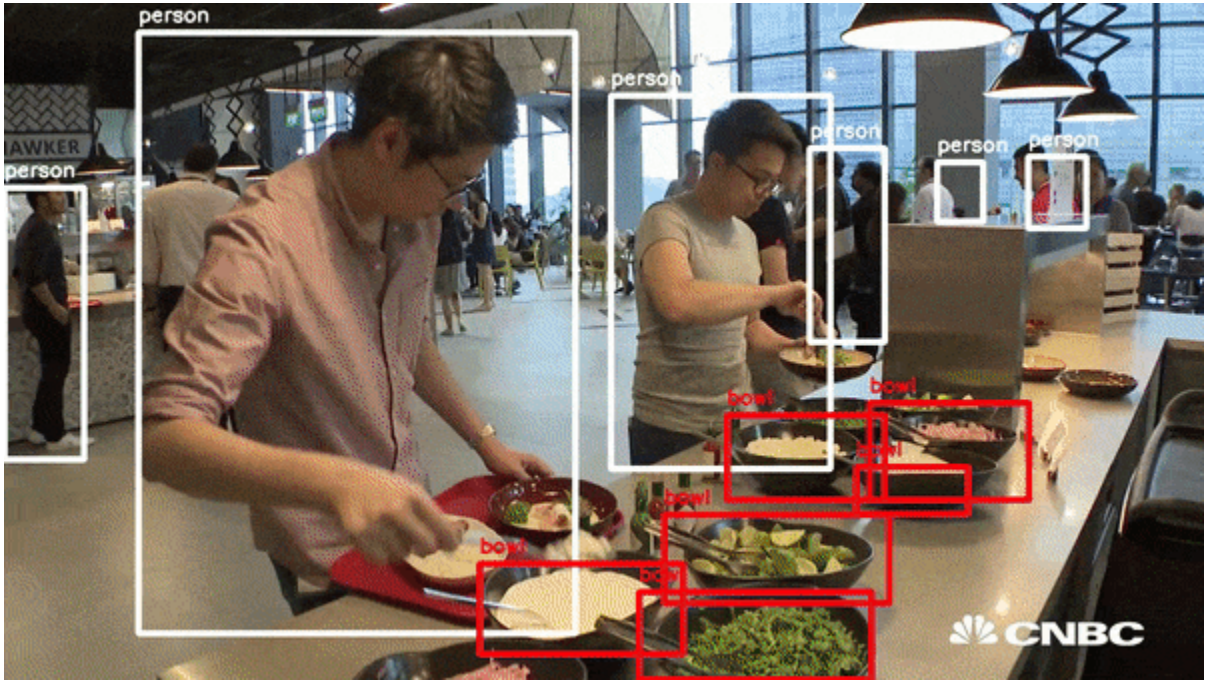
- Sentiment classification



Reference: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.703.6858&rep=rep1&type=pdf

More Application: YOLO

- YOLO is real-time object detection and classification.



https://github.com/thtrieu/darkflow

More Application: Google Quick Draw

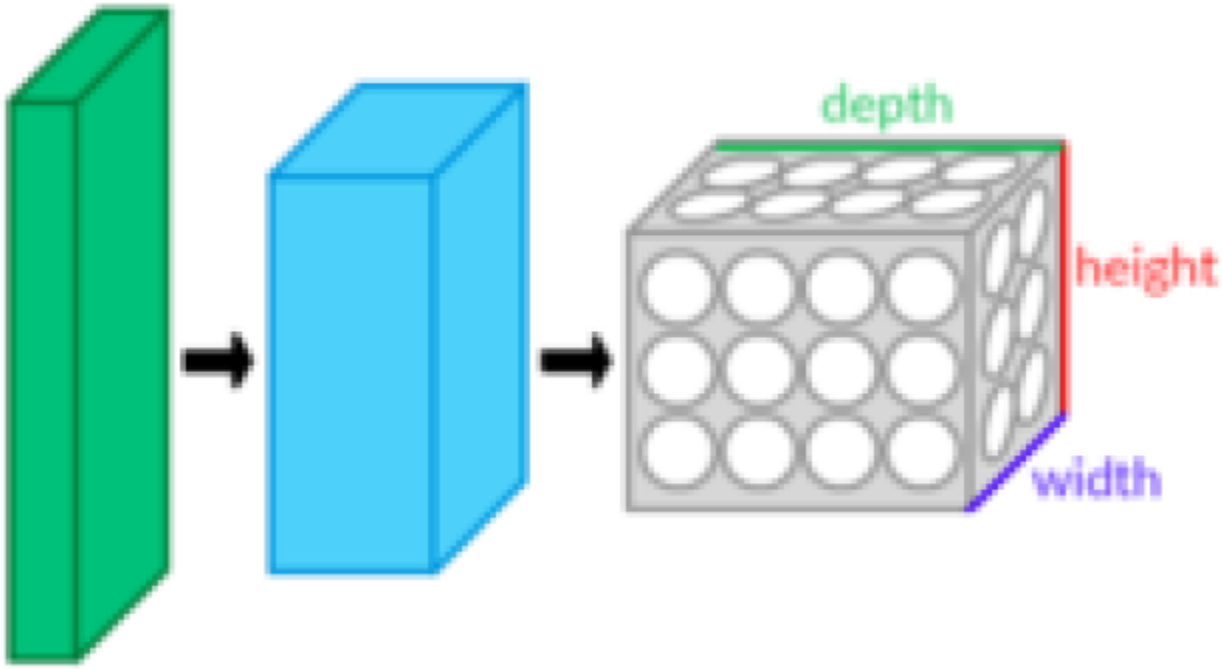
- https://quickdraw.withgoogle.com/

類神經網路能學會辨識塗鴉嗎？

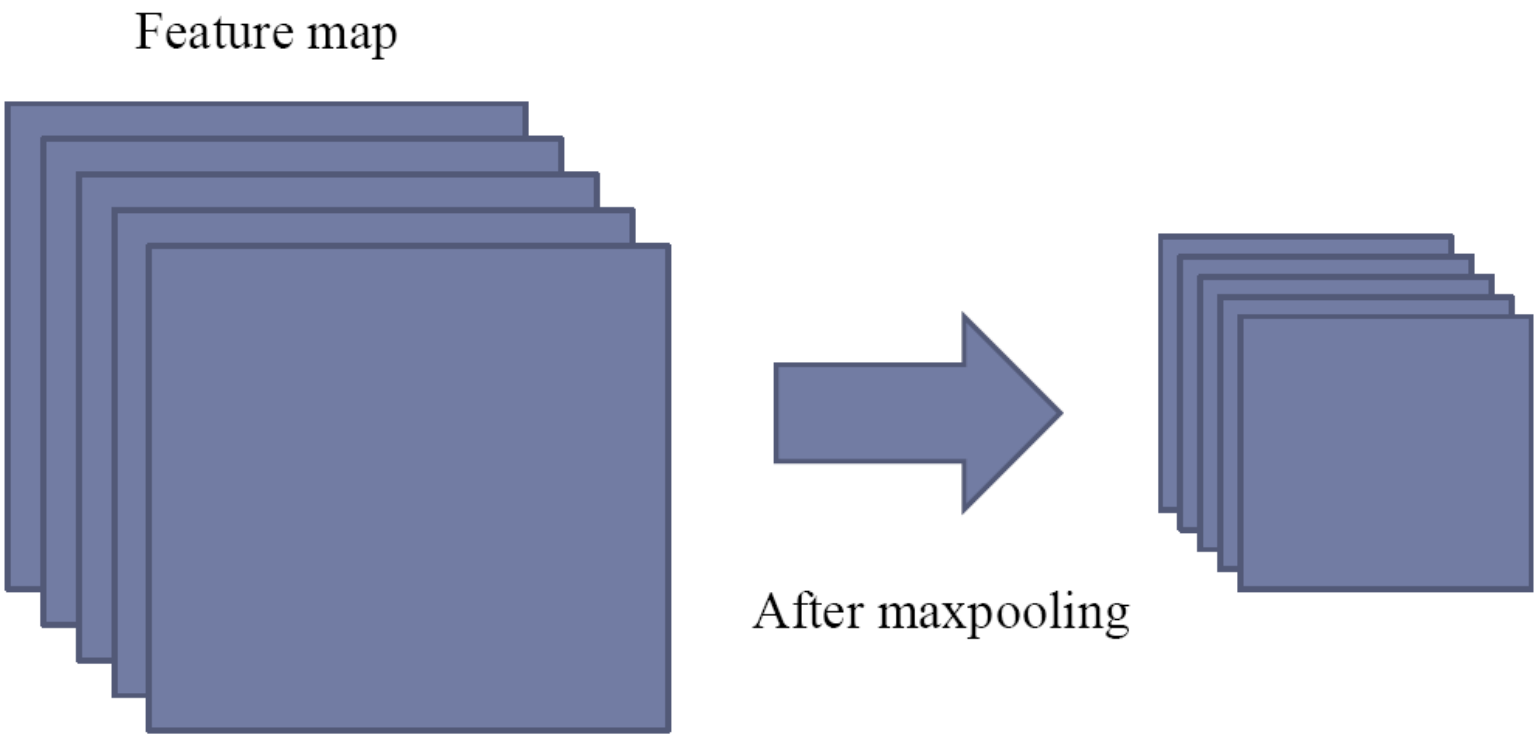
只要將你的繪圖加到全世界最大的塗鴉資料集，就能協助訓練這個類神經網路。這個資料集的內容會公開分享，為機器學習研究提供參考資料。

Summary

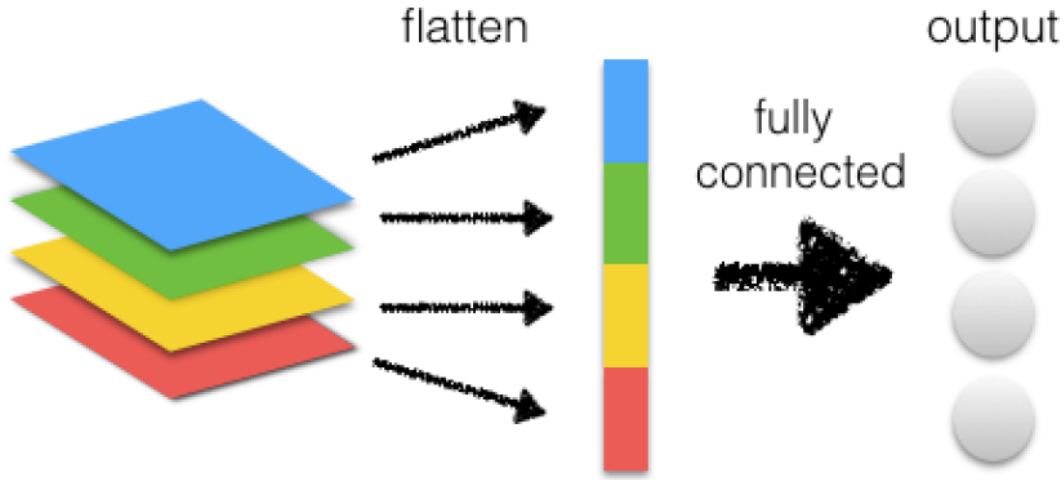
Summary: CNN Convolution Layer

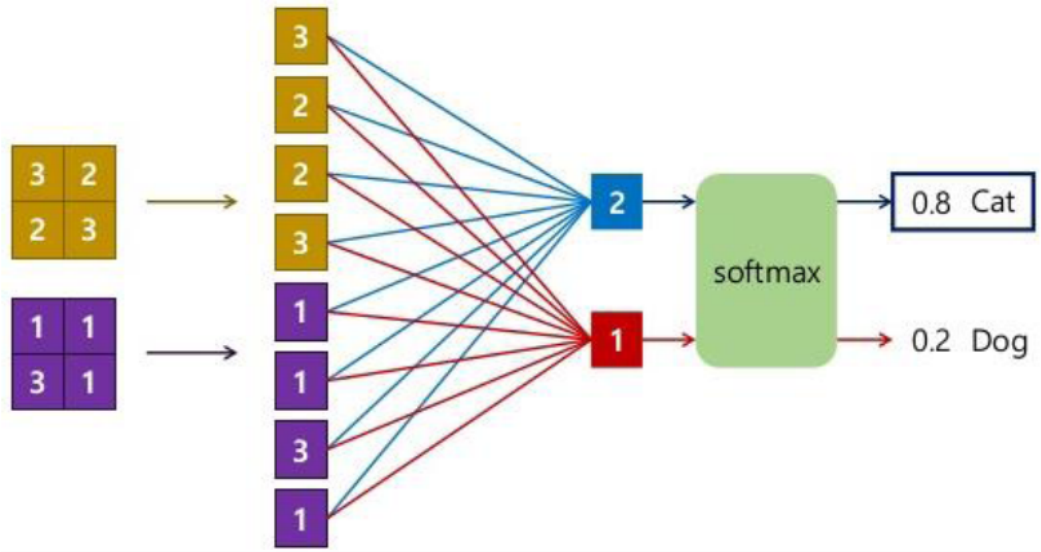


Summary: CNN Max Pooling Layer

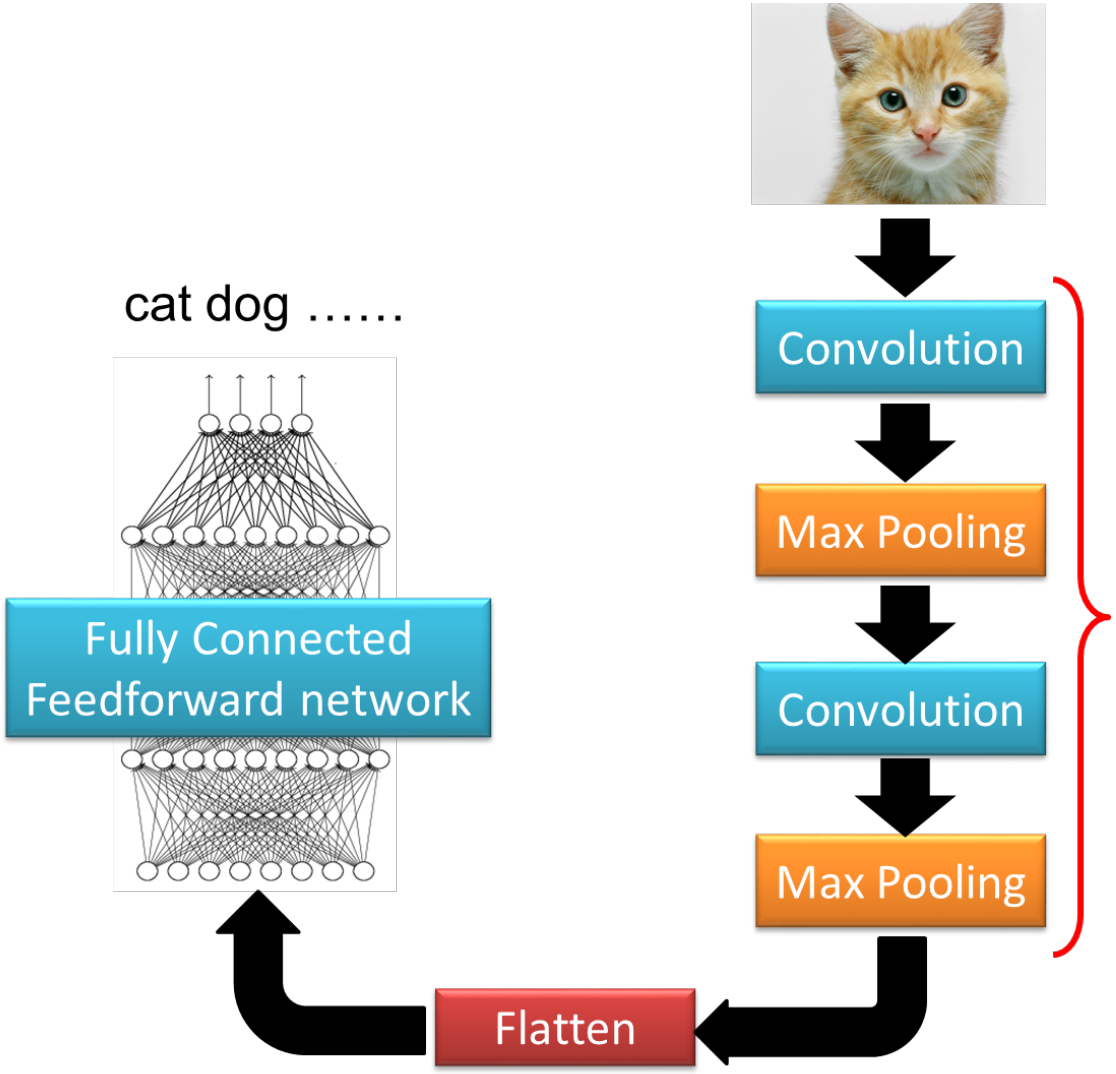


Summary: Flatten





Summary: CNN


cat dog ……
Fully Connected Feedforward network
Convolution
Max Pooling
Convolution
Max Pooling
Flatten

To learn more ……

- The methods of visualization in these slides
  - https://blog.keras.io/how-convolutional-neural-networks-see-the-world.html
- More about visualization
  - http://cs231n.github.io/understanding-cnn/
- Very cool CNN visualization toolkit
  - http://yosinski.com/deepvis
  - http://scs.ryerson.ca/~aharley/vis/conv/
- The 9 Deep Learning Papers You Need To Know About
  - https://adeshpande3.github.io/adeshpande3.github.io/The-9-Deep-Learning-Papers-You-Need-To-Know-About.html

To learn more ……

- How to let machine draw an image
- PixelRNN
  - https://arxiv.org/abs/1601.06759
- Variation Autoencoder (VAE)
  - https://arxiv.org/abs/1312.6114
- Generative Adversarial Network (GAN)
  - http://arxiv.org/abs/1406.2661